Ontology-Based Legal Information Extraction

Group 27

Keet Sugathadasa (130581H)
Vindula Jayawardana (130247P)
Buddhi Ayesha (130508T)
Dimuthu Lakmal (130323V)

Supervised by:
Dr. Shehan Perera
Mr. Nisansa de Silva

Department of Computer Science & Engineering

Ontology-Based Legal Information Extraction

Group 27

by

Keet Sugathadasa (130581H)
Vindula Jayawardana (130247P)
Buddhi Ayesha (130508T)
Dimuthu Lakmal (130323V)

Project duration:	January 1, 2017 – January 1, 2018
Supervised by:	Dr. Shehan Perera
	Mr. Nisansa de Silva

Contents

1 Introduction

Reading and understanding a given piece of text is merely a walk in the park for a human being, if he or she is aware of the context of the text. This understanding process is supported by past experiences, domain knowledge and the structure of the given piece of text. With this understanding, a person could draw related conclusions to support the problem at hand. But this process is not as easy for computers, as it appears to be for humans. Drawing the conclusion to a given problem using the information and knowledge at hand, has always been a problem, in terms of efficiency, accuracy, and reliability. The information for a given domain increases day by day, making the entire process even more complex. Due to this reason, scientists have tried to come up with methods to get the necessary information as required from a given source, and this process is known as information extraction.

In Artificial Intelligence, information extraction is the drawing out of certain types of information from mostly natural language text, by processing them automatically [1]. Approaches like linguistic rules, gazetteer lists and web based search, are being used to carry out information extraction. Information extraction lies in between text understanding systems that attempt to analyze text and extract their semantic contents, and information retrieval systems which is merely a retrieval of information and documents related to the user's requirements [1]. In contrast, information extraction extracts information from natural language texts.

Given a domain of interest, it is necessary to generate an ontology or a knowledge base that would support the purpose of information extraction. "An ontology is an explicit specification of a conceptualization of a given domain" [2]. The use of ontologies for information extraction is known as ontology based information extraction. These systems may construct ontologies (by identifying concepts and relationships) and carries out the information extraction from necessary sources. An ontology is a meta-level description of the knowledge base of the domain of interest, where the ontology becomes a reusable component across that domain.

In the case of legal domains, "there is a massive legal literature in printed documents as well as online, where the current accessibility of sources of law does not suffice to serve legal and non-legal professionals" [3]. Implementing a reliable system that would be able to extract legal information from these online sources automatically, would be beneficial to legal officers and the general public as well. On the contrary, unlike in other domains, some of the legal case structures differ significantly [4], where special attention is needed to be given to the legal domain in the process of information extraction. One approach to solve this problem would be the use of ontologies in the legal domain. The generation of a legal ontology is not an easy task due to the various structures of cases and norms available in the domain. This difficulty can be mitigated by integrating domain knowledge into the ontology generation process. Afterward, the generated ontology can be used to guide the extraction of necessary information through an ontology based information extraction process. Having a user query system integrated with natural language processing to carry out the legal information extraction process, where the user simply has to enter the problem in natural text, would be an ideal solution to enhance the ontology based legal information extraction systems in the legal domain to provide easy interoperability and accessibility.

1.1. Problem Statement

The lack of a sophisticated system for lawyers, paralegals and the general public to get relevant legal information (related court cases, relationships among cases, judgments, statutes and keyword definitions) regarding a new court case of interest, simply by entering a legal scenario in natural text without the need of having a broad knowledge about the legal domain.

1.2. Motivation

When a lawyer or a paralegal is faced with a court case or a similar task, he or she will have to go through a lot of documents by researching on the relevant case to get the necessary information needed for that case. This information may consist of different aspects, such as previous court cases, judgments and laws which will have a major impact on the given case. However, due to the way that this information has been stored, legal officers find it extensively painful to carry out manual searching, for the relevant information of a given case. Apart from that, this manual process is subjected to erroneous situations where a small human error could make a huge impact on a given case. To avoid those detrimental scenarios, online systems and frameworks like WestLaw and LexisNexis came into play. However, due to the fact that the aforesaid systems only provide query based searching, where legal officers need to remember keywords which are predefined when querying for relevant legal information, there is still a hassle in accessing this information. There is no simple and easy way for lawyers to enter some natural text on a given case and get the information relevant to that case effectively and reliably.

This raises a potential need for a proper system to do hassle free legal information extraction based on natural text queries. While there have been some attempts built on rule based, case based and statistics based methods, to overcome the aforesaid requirement, Araujo et al [5] have addressed this need in an ontology based simulated environment, which had created a novel way of achieving the given goal. Due to their experiment on a limited number of case documents in Brazil, this has created a requirement for in-depth research on the aforesaid aspect. Hence, there is a potential research gap for a proper ontology based legal information extraction system, to make the paralegal's life easy.

It should also be noted that access to "legal information and legal literature is a fundamental democratic right" of all citizens of a country irrespective of the fact that they have had a legal education or not. Having that kind of a legal background, arises the need for a proper system which allows any citizen to access the legal information, irrespective of his or her education level.

The system that is proposed by this proposal, is to cater the aforementioned aspects, by creating a platform which does effective and reliable legal information extraction based on natural language text. By eliminating these current limitations, it would allow any legal officer to effectively access the relevant information. In addition to that, it would also allow any individual to access the legal information irrespective of his or her education level and ability to form a specific type of query.

1.3. Research Objectives

The proposed Ontology Based Legal Information Extraction System, consists of the following sub components.

- Web-Crawler
- Preprocessor
- Legal Ontology
- Ontology Knowledge Base
- Information Extraction Module
- Information Retrieval Module
- Accuracy Measuring System
- Query Answering Module

Following are the main research components of this project. These research topics represent a sequence of studies that needs to be carried out, in order to finalize the product accordingly. This also includes researches with specific accuracy measurements, in order to validate the study and usage of certain tools and techniques for this research. Some of these topics are explained below.

1. Web Crawling from Online Repositories
2. Known Approaches for Information Extraction
3. Generate Ontology with Legal Domain Experts
4. Ontology Population with Information Extraction
    - Ontology class vector representation
    - Semi supervised Ontology Population
5. Legal Case Document Vector Representation
    - Similarity Based Word Vector Embeddings
    - Ranking based legal case vector representation
6. Legal Document Retrieval Accuracy Measuring System
7. Natural Text to Ontology Based Query Mapping
8. Query to Retrieval, Accuracy Measuring System

1.3.1. Similarity Based Word Vector Embeddings

Semantic Similarity measurements based on linguistic features are a fundamental component of almost all Natural Language Processing (NLP) tasks: Information Retrieval, Information Extraction, and Natural Language Understanding [1]. In the case of NLP based Information Retrieval (IR), it plays into the task of obtaining the items that are most relevant to the query whereas Information Extraction (IE), plays into the task of correctly recognizing the linguistic elements to be extracted be it the Part of Speech (PoS) tags or Named Entities in Named Entity Recognition (NER). In the case of Text Understanding which is also known as Natural Language Understanding (NLU), it helps in identifying semantic connections between elements on the document that is being analyzed.

Law and order could be rather regarded as the cloak of invisibility that operates and controls the human behavior to its possible extents in the name of justice. Thus in terms of maintaining social order, quiddity of law within the society is mandatory. John Stuart Mill articulated a principle in On Liberty, where he stated that *The only purpose for which power can be rightfully exercised over any member of a civilized community, against his will, is to prevent harm to others* [6]. Being such a vital field, acquisition of laws and legal documents through technological means is on the list of necessities on the rise. Also, ample justification for the need of semantic disambiguation in the legal domain can be found in seminal cases such as: *Fagan v MPC* and *R v Woollin*. Therefore we selected the law as the domain for this study.

The legal domain contains a considerable amount of domain specific jargon where the etymology lies with mainly Latin and English. To complicate this fact, in certain cases, the meaning of the words and context differs by the legal officers interpretations.

Another field which suffers similarly from this issue is the medical industry [7]. Non-systematic organization and complexity of the medical documents, result in medical domain falling victim to loss of having proper representations for its terminology. PubMed[1] attempts to remedy this. Later studies such as [8] utilize these repositories. Therefore, it is possible to claim that the problem addressed in this study is one that is not just limited to the legal domain but is one that transcends into a numerous other domains as well.

Methods that treat words as independent atomic units are not sufficient to capture the expressiveness of language [9]. A solution to this is word context learning methods [10, 11]. Another solution is lexical semantic similarity based methods [12]. Both of these approaches try to capture semantic and syntactic information of a word. In this study we propose a methodology to have a synergistic union of both of these methods. First for word context learning, we used a Word Embedding method, word2vec [9, 13, 14]. Then we used a number of lexical semantic similarity measures [15–17] to augment and improve the result.

The hypothesis of this research has three main claims: (1) A word embedding model trained on a small domain specific corpus can outperform a word embedding model trained on a large but generic corpus, (2) Word lemmatization, which removes inflected forms of words, would improve the performance of a word embedding model, (3) Usage of lexical semantic similarity measures trained over a machine learning system can improve the overall system performance. Our results sufficiently prove all of these claims to be true.

1.3.2. Ontology class vector representation

Semantic models are used to present hierarchy and semantic meaning of concepts. Among them ontologies are a widely used superlative model extensively applied to many fields. As defined by Thomas R. Gruber [18], an ontology is a “formal and explicit specification of a shared conceptualization". The use of ontologies is becoming increasingly involved in various computational tasks given the fact that ontologies can overcome limitations in traditional natural language processing methods in domains such as text classification [12, 19], word set expansions [20], linguistic information management [21, 22], medical information management [23, 24], and Information Extraction [1, 8].

However very few attempts have been made on representing ontology classes in different representations such as vector representations. The importance of having different representations for ontology classes is emphasized when it comes to ontology mapping, ontology merging, ontology integration, ontology alignment and semi automated ontology population. However sophisticated researches on representing ontology classes in different representations is still an open ended question.

In this study we propose a novel way of deriving representative vectors for ontology classes. This is an important problem in the domain of automatic ontology population and automatic ontology class labeling. We use a distributed representation of words in a vector space grouped together [25], achieved by means of word vector embeddings to transform the word strings in instances and the class labels to the same vector space. For this task of word embedding, we chose the neural network based method: word2vec, proposed by Tomas Mikolov et al. [9], which is a model architecture for computing continuous vector representations of words from very large data sets.

In the proposed methodology we created an ontology in the domain of consumer protection law with the help of legal experts. The word2vec model was trained with the legal cases from FindLaw [2] online database. The word embedding vectors of the instances and the class labels in the created ontology were the obtained using the trained word2vec model. For each ontology class a number of candidate vectors were then calculated using the word embedding vectors of the instances. The candidate vectors and the class label vectors were then used to train a machine learning model to predict the best representative vector. We show that our proposed methodology outperforms the traditional average (mean) vector representation and median vector representation in all classes. On average, the distance of our representative vector to the class vector is 0.82, while the mean vector has a distance of 1.31 and the median vector has a distance of 1.51. Respectively, it is a 37% and 50% improvement.

[1]https://www.ncbi.nlm.nih.gov/books/NBK3827/

[2]http://caselaw.findlaw.com/

1.3.3. Ontology Population with Information Extraction

In various computational tasks in many different fields, the use of ontologies is becoming increasingly involved. Many of the research areas such as knowledge engineering and representation, information retrieval and extraction, and knowledge management and agent systems [26] have incorporated the use of ontologies to a greater extent. As defined by Thomas R. Gruber [18], an ontology is a “formal and explicit specification of a shared conceptualization”. Due to the evolving ability of ontologies to overcome limitations in traditional natural language processing methods, the popularity of using ontologies in modern computation tasks are getting increased day by day. For an example, text classification [12, 19], word set expansions [20], linguistic information management [21, 22, 27, 28], medical information management [23, 24], and information extraction [1, 8] emphasize the growing popularity of the ontology based computations and processing.

According to Carla Faria et al. [29], ontology population looks for instantiating the constituent elements of an ontology, like properties and non-taxonomic relationships. However, most of the time, ontology populations are done by domain experts and knowledge engineers as a manual process, which is both time consuming and expensive. As majority of the world’s knowledge is encoded in natural language text, automating the population of these ontologies using results obtained from Natural Language Processing (NLP) based analysis of documents, has recently become a major challenge for NLP applications [30].

In this study, we propose a novel way for semi-supervised instance population of an ontology using word vector embeddings. Word Embeddings could be identified as a collective name for a set of language modeling and feature learning techniques in natural language processing. The basic idea behind word embedding is based on the concept where words or phrases from the vocabulary are mapped to vectors of real numbers. We use these vectors as a method of arriving at instance population in an ontology. For this purpose, we built an iterative model based on the class representative vector for ontology classes [13]. In our implementation, we built multiple models based on different methodologies. In one model we assigned membership to natural language tokens by distance to the representative vectors. In another, we used word2Vec’s internal dissimilar exclusion method to identify the membership. In another model, we used set expansion as described by [20], for the purpose of ontology population. As each model outputs a set of candidate words for a given class, we then collaborate with domain experts and knowledge engineers to identify the performance of each model.

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). It has been observed that many machine leaning approaches elucidate considerable improvement in learning accuracy, when unlabeled data is used in conjunction with a small amount of labeled data.

The legal context contains jargon which is complex and most of the time impossible to have stored in mind; whether it be an average person or a paralegal, given that it consists terminology derived from ancient Latin terms, as well as various distinctive terminology depending on the category of laws and the geographical settings of practice. Therefore, knowing them manually is rather an impossible task which drove us to select the legal domain for this study of semi-supervised ontology population.

1.3.4. Legal Document Retrieval using document vector embeddings

Similarity measures between words, sentences, paragraphs, and documents are a prominent building block in majority of tasks in the field of Natural Language Processing. Information Retrieval (IR) is an application that heavily uses similarity measures. This is due to the fact that the function of IR algorithms require the identification of similarity and context of interest. Thus when the IR is done on textual documents, semantic similarity measures play a major role in both identifying documents related to the query and ranking the resultant documents according to the relevance [31].

Even though document retrieval is a well researched and mature area of study, it becomes a different and unique problem for each domain of research. This is mainly due to the syntactical complexities in the textual documents and the semantic structure of the text. Textpresso [32] is a text mining, information extraction, and information retrieval system that goes far beyond traditional keyword search engines, built for biological literature. Further in the biological domain, OmniSearch [23] is a semantic search system based on the ontology for microRNA-target gene interaction data [24]. All other fields such as medicine [7], geology [33], and music [34] also have their own unique aspects which make the information retrieval task more complex and domain specific.

In terms of carrying out a legal case in particular and also as the courts are binding upon precedent to know the law and knowing under which case that the laws were established and enforced is of utmost importance. There are also instances of which courts turn into case law of other countries in the absence of their own that explains the case context at hand. Amongst the large number of cases and the phases of which the cases have evolved, it remains impossible for those in the field of law to remember and have cases and laws in memory.

Because of these reasons, we selected the legal document retrieval as our domain. The legal domain, which is our primary focus in this study, contains considerable amount of domain specific jargon where the etymology lies with mainly Latin and English, which makes the information retrieval (IR) task multilingual. To complicate this fact, in certain cases, the meaning of the words and context differs by the legal officers' interpretations. This is the main standout for current legal IR systems such as Westlaw[3] and LexisNexis[4]. The main drawback of these systems is that, they still being boolean indexed systems, extensive user training is required of a legal professional to utilize them [35]. Despite this shortfall, Westlaw and LexisNexis have the largest number of paying subscribers for legal information retrieval, which indeed validates the need for a legal information retrieval system..

This study targets the information retrieval for the legal domain where experiments are being carried out over 2500 legal cases collected from Findlaw [36] and other online legal resources via both retrieval and extraction. We propose a system that includes a page ranking graph network with TF-IDF to build document embeddings by creating a vector space for the legal domain, which can be trained using a neural network model supporting incremental and extensive training for scalability of the system.

[3]https://www.westlaw.com/
[4]https://www.lexisnexis.com/

2 System Requirement Specification

2.1. Introduction

This proposed system is primarily based for the legal domain, which addresses the day to day problem every legal officer faces, with respect to the legal domain. This is mainly due to the difficulty in finding relevant legal cases, legal information and related court cases, that match the legal officer's requirements. The system proposed in this study is an Ontology Based Legal Information Extraction System, that is capable of retrieving and extracting relevant legal information according to a given use query. This system also caters to the general public where they are given access to the system, to search any legal information they need.

Sections given below in this chapter addresses the requirements of the system in a product scope

2.1.1. Purpose

The System Requirements Specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide. This is one of the major artifacts produced to clients and managers, to take proper decisions based on the product and development. This document outlines the entire system with its scope, functional requirements, non functional requirements, and also the necessary details related to the final product.

The first part of this document establishes a well-defined platform, in terms of the system requirements for the development and implementation of the product. This document specifies all the related use cases of the entire system with the detailed sequence of operation of each use case and their constraints on operation. Apart from that, this provides a comprehensive analysis on the nonfunctional requirements needed for the system with the possible and existing design constraints.

Second part of the document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

Following sections illustrate architectural decisions made on the system in terms of the use cases and other perspectives. The document is intended for the use in development and decision making, for both the developers and project clients.

2.1.2. Scope

The system has a very specific set of use cases, where it provides the general functionality to cater to the problem being addressed. It limits to the following use cases, listed below. These outline the general functionality of the system, which can be utilized by the stakeholders as mentioned in Section 2.1.3.

1. View Legal Ontology
2. Search Legal Ontology
3. Query system using natural language text

4. View Details of the system
5. View Summary and Detailed Legal Cases

The artifacts provided by this system, for its stakeholders are as follows.

- Legal Ontology
- Website
- Query System

2.1.3. Direct Stakeholders

Given below, are the major stakeholders of this system. The scope mentioned in Section 2.1.2, will be primarily accessed by these stakeholders.

- Legal Professionals
- General Public

2.1.4. Definitions, Acronyms, and Abbreviations

- OBLIE : Ontology Based Legal Information Extraction System
- OBIE : Ontology Based Information Extraction System
- IE : Information Extraction
- SRS : System Requirement Specification

2.1.5. Overview

In this system requirement specification document, the following major concerns have been comprehensively addressed. These are primarily targeting the stakeholders mentioned in Section 2.1.3, where the following features will be a greater support to engage in their daily activities.

- Specific requirements with detailed descriptions for designing
- Functional Requirements of the system
- Non Functional Requirements of the system
- System architecture specification
- Supportive information for ease of use of the document

2.2. Specific Requirements

This section provides the major requirements being addressed in this section, and explains each of the use cases for the reader to get a clear understanding about the system.

2.2.1. Functional Requirements

A. View Legal Ontology

This provides a comprehensive and validated legal ontology that will provide the concepts and their relationships between the concepts. The user can get a clear idea about the legal domain and its inside relationships.

B. Search Legal Ontology

This provides a search facility on the Ontology, where the user can search for concepts and relationships as needed. This also facilitates features like viewing metadata and filtering out certain levels of the ontology as needed, to reduce the complexity of the viewing system.

C. Query system using natural language text

In this use case, the user will use the natural human language (English) to specify the case at hand (information need). The system will respond to that by retrieving the documents relevant to the user's query. The user may enter the required user story and the system will provide the most relevant set of cases to the user.

D. View Details of the System

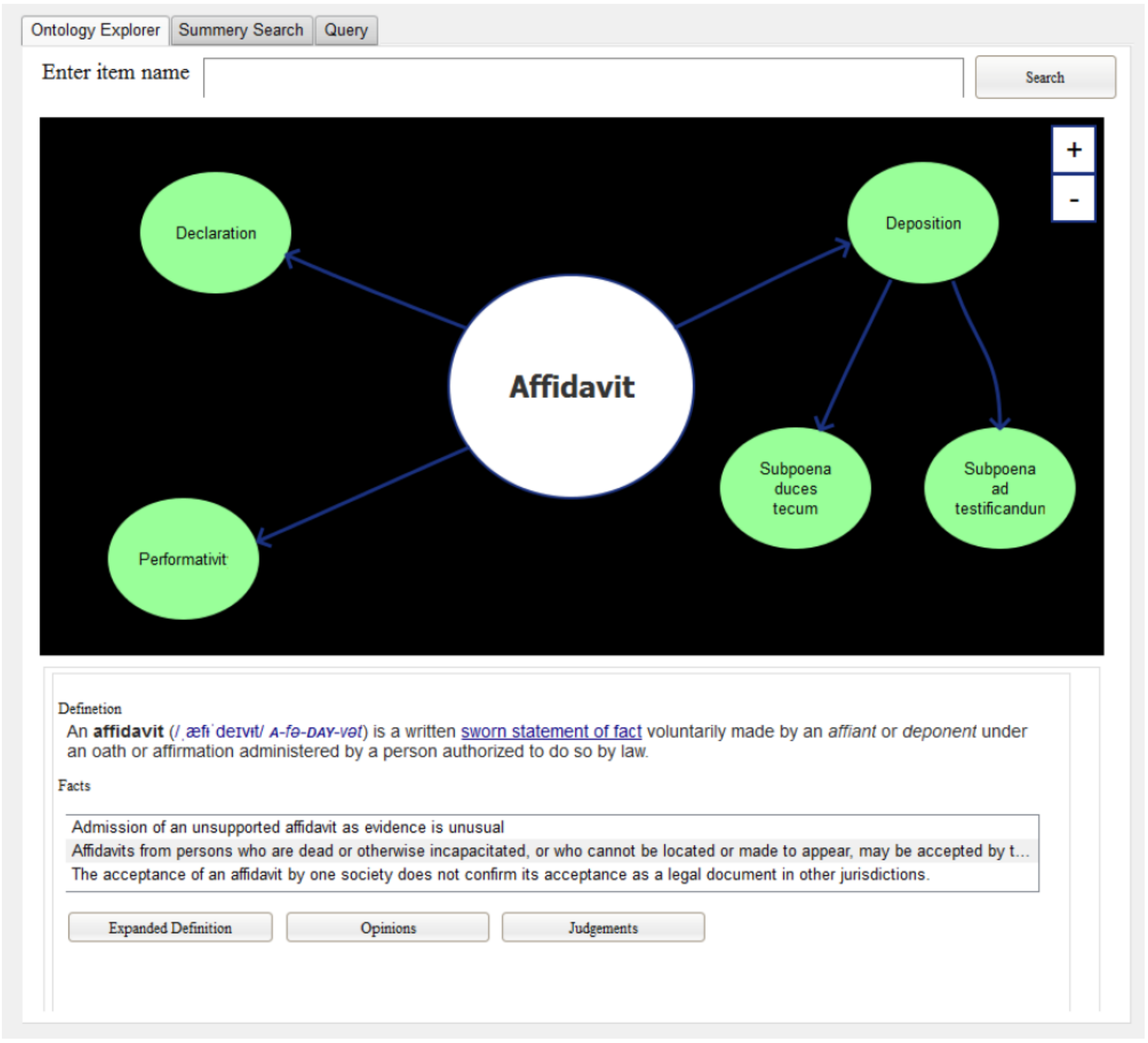
This capability is provided via the main website of the system which provides necessary details about the functionality, packages, user stories and other relevant details of the system. This provides as an entry point to users, where they can access other functionalities of the system.

E. View Summary and Detailed Legal Cases

This use case can be run in isolation or as an extension to the querying use case. This provides the ability for the user to view the legal cases as summaries or as detailed descriptions that provide the final result of the user need.

2.2.2. Interfaces

Following two prototypes illustrate the mock-up interfaces of the system, in Figure 2.1 and Figure 2.2.



Defination

An **affidavit** (/ˌæfɪˈdeɪvɪt/ *A-fə-DAY-vət*) is a written sworn statement of fact voluntarily made by an *affiant* or *deponent* under an oath or affirmation administered by a person authorized to do so by law.

Facts

Admission of an unsupported affidavit as evidence is unusual
Affidavits from persons who are dead or otherwise incapacitated, or who cannot be located or made to appear, may be accepted by t...
The acceptance of an affidavit by one society does not confirm its acceptance as a legal document in other jurisdictions.

Figure 2.1: Query Answering Interface 1

Shown in Figure 2.3 and Figure 2.4 are some examples of the system, that will be presented to the user.

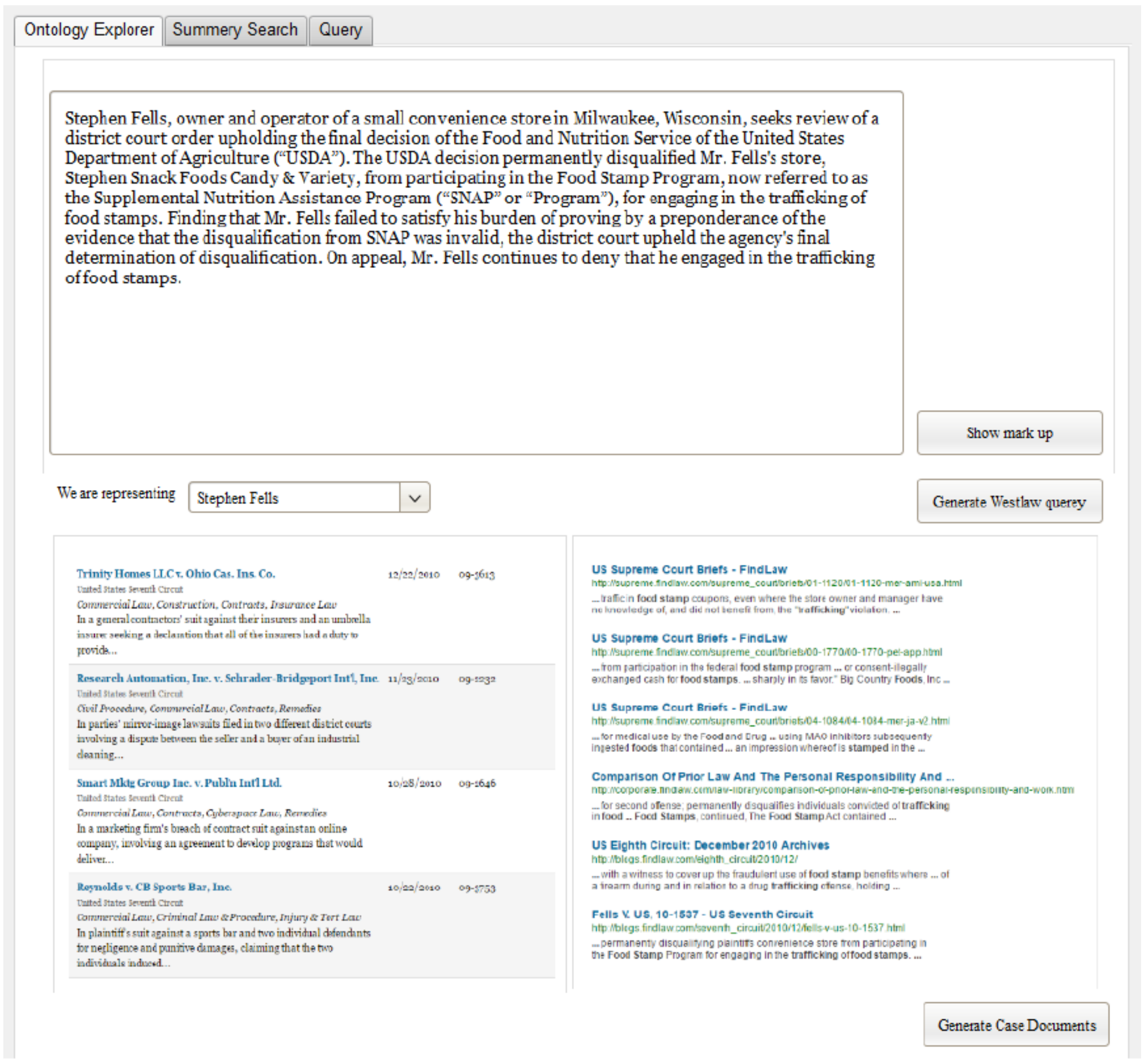
2.3. Architectural Representation

Table 2.1: Distance measures of the classes and the average over 10 classes

View	Representation	Description
Use case view	Use case diagram	This captures the major use cases of the system
Logical View	Class Diagram	This captures all the domain level classes and their associations
Process View	Activity diagram	These captures all the use case flows of operation.
Data View	ER diagram	This illustrates the entities, relationship sets and their associations

2.3.1. Use-Case View

Figure 2.3.1 illustrates the use cases of the system, with the relevant stakeholders and their interaction with the system in terms of use cases. These are the use cases mentioned in Section 2.2.1 and the stakeholders are mentioned in Section 2.1.3. This is depicted in Figure 2.5.



Stephen Fells, owner and operator of a small convenience store in Milwaukee, Wisconsin, seeks review of a district court order upholding the final decision of the Food and Nutrition Service of the United States Department of Agriculture ("USDA"). The USDA decision permanently disqualified Mr. Fells's store, Stephen Snack Foods Candy & Variety, from participating in the Food Stamp Program, now referred to as the Supplemental Nutrition Assistance Program ("SNAP" or "Program"), for engaging in the trafficking of food stamps. Finding that Mr. Fells failed to satisfy his burden of proving by a preponderance of the evidence that the disqualification from SNAP was invalid, the district court upheld the agency's final determination of disqualification. On appeal, Mr. Fells continues to deny that he engaged in the trafficking of food stamps.

We are representing: Stephen Fells

Trinity Homes LLC v. Ohio Cas. Ins. Co. 12/22/2010 09-3613
United States Seventh Circuit
Commercial Law, Construction, Contracts, Insurance Law
In a general contractors' suit against their insurers and an umbrella insurer seeking a declaration that all of the insurers had a duty to provide...

Research Automation, Inc. v. Schrader-Bridgeport Int'l, Inc. 11/23/2010 09-1232
United States Seventh Circuit
Civil Procedure, Commercial Law, Contracts, Remedies
In parties' mirror-image lawsuits filed in two different district courts involving a dispute between the seller and a buyer of an industrial cleaning...

Smart Mktg Group Inc. v. Publ'n Int'l Ltd. 10/28/2010 09-1646
United States Seventh Circuit
Commercial Law, Contracts, Cyberspace Law, Remedies
In a marketing firm's breach of contract suit againstan online company, involving an agreement to develop programs that would deliver...

Reynolds v. CB Sports Bar, Inc. 10/22/2010 09-3753
United States Seventh Circuit
Commercial Law, Criminal Law & Procedure, Injury & Tort Law
In plaintiff's suit against a sports bar and two individual defendants for negligence and punitive damages, claiming that the two individuals induced...

US Supreme Court Briefs - FindLaw
http://supreme.findlaw.com/supreme_court/briefs/01-1120/01-1120-mer-ami-usa.html
... trafficin **food stamp** coupons, even where the store owner and manager have no knowledge of, and did not benefit from, the "**trafficking**" violation. ...

US Supreme Court Briefs - FindLaw
http://supreme.findlaw.com/supreme_court/briefs/00-1770/00-1770-pet-app.html
... from participation in the federal **food stamp** program ... or consent-illegally exchanged cash for **food stamps**. ... sharply in its favor." Big Country **Foods**, Inc ...

US Supreme Court Briefs - FindLaw
http://supreme.findlaw.com/supreme_court/briefs/04-1084/04-1084-mer-ja-v2.html
... for medical use by the Food and Drug ... using MAO inhibitors subsequently ingested **foods** that contained ... an impression whereof is **stamped** in the ...

Comparison Of Prior Law And The Personal Responsibility And ...
http://corporate.findlaw.com/law-library/comparison-of-prior-law-and-the-personal-responsibility-and-work.htm
... for second offense; permanently disqualifies individuals convicted of **trafficking** in **food** ... **Food Stamps**, continued, The **Food Stamp** Act contained ...

US Eighth Circuit: December 2010 Archives
http://blogs.findlaw.com/eighth_circuit/2010/12/
... with a witness to cover up the fraudulent use of **food stamp** benefits where ... of a firearm during and in relation to a drug **trafficking** offense, holding ...

Fells V. US, 10-1537 - US Seventh Circuit
http://blogs.findlaw.com/seventh_circuit/2010/12/fells-v-us-10-1537.html
... permanently disqualifying plaintiffs convenience store from participating in the **Food Stamp** Program for engaging in the **trafficking** of **food stamps**. ...

Figure 2.2: Query Answering Interface 2

2.3.2. Logical View

The following section describes the domain level overview of the entire system. The whole system domain, is abstracted in the following section with relevant information. Figure 2.6 is the class diagram of the system.

2.3.3. Process View

This section is intended to illustrate the main modes of communication between processes. Figure 2.7, Figure 2.8, and Figure 2.9 describes the activity diagrams of the most critical use cases of the system.

2.3.4. Data View

Figure 2.10, Figure 2.11, Figure 2.12, Figure 2.13, Figure 2.14, Figure 2.15, Figure 2.16, and Figure 2.17, describes the entity relationship diagrams of the system with respect to the data view concerns.

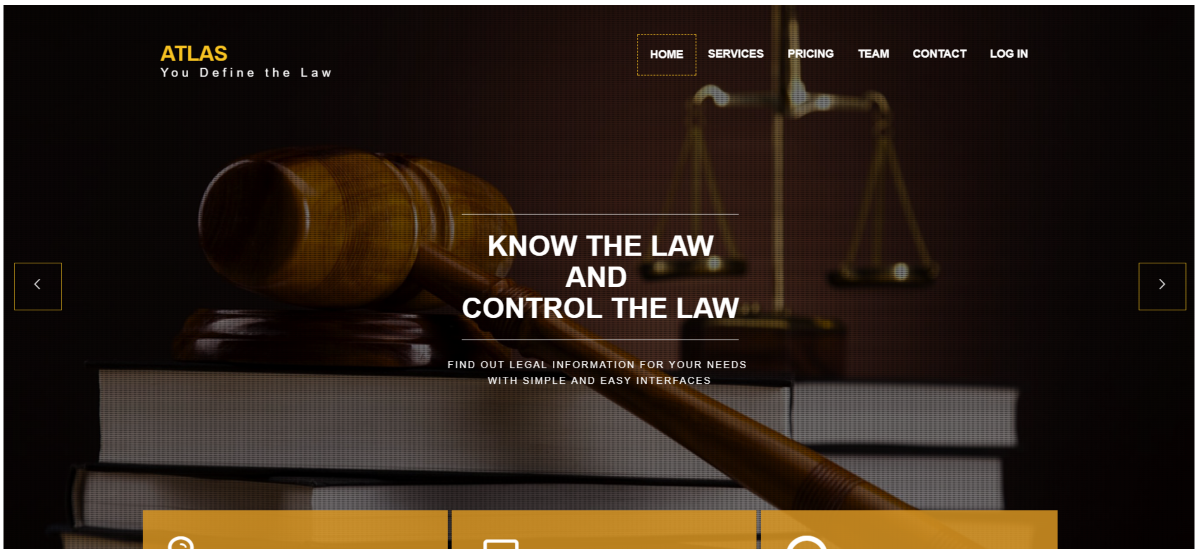
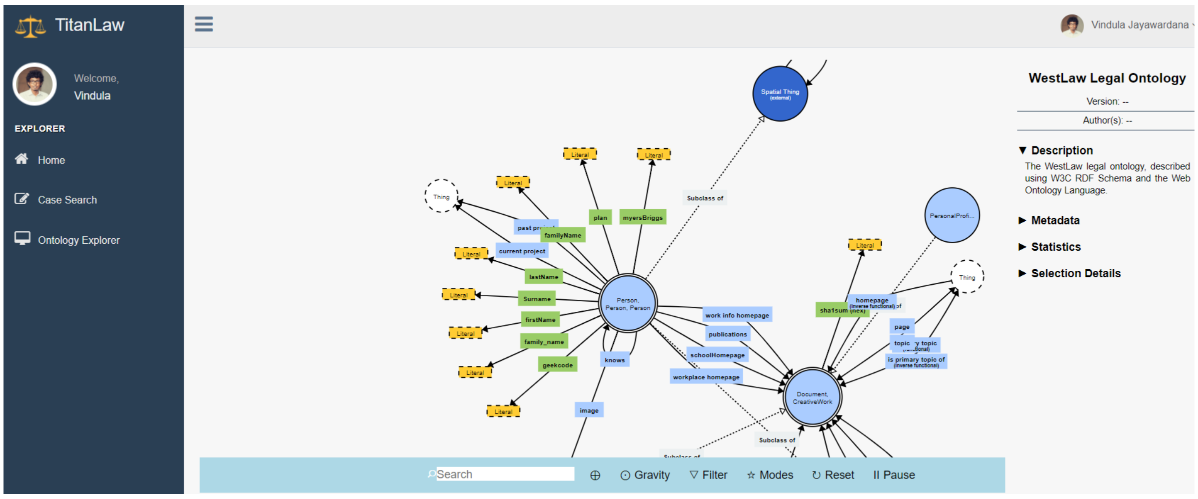

ATLAS
You Define the Law
HOME
SERVICES
PRICING
TEAM
CONTACT
LOG IN
KNOW THE LAW AND CONTROL THE LAW
FIND OUT LEGAL INFORMATION FOR YOUR NEEDS WITH SIMPLE AND EASY INTERFACES

Figure 2.3: Front View of the Website



WestLaw Legal Ontology

Version: --

Author(s): --

▼ Description

The WestLaw legal ontology, described using W3C RDF Schema and the Web Ontology Language.

▶ Metadata

▶ Statistics

▶ Selection Details

Figure 2.4: Ontology Explorer

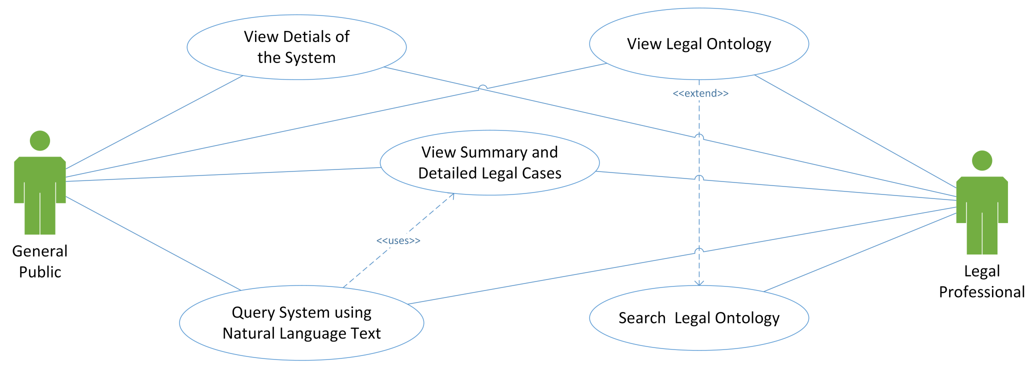

View Detials of the System
View Legal Ontology
<<extend>>
View Summary and Detailed Legal Cases
<<uses>>
Query System using Natural Language Text
Search Legal Ontology
General Public
Legal Professional

Figure 2.5: Use Case Diagram

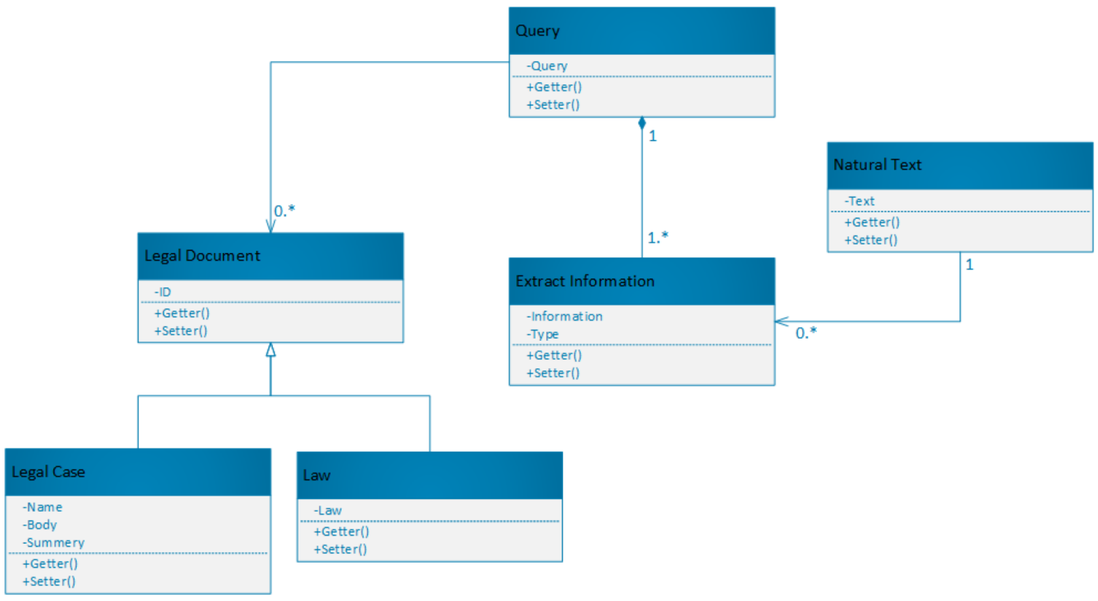


Figure 2.6: Class Diagram

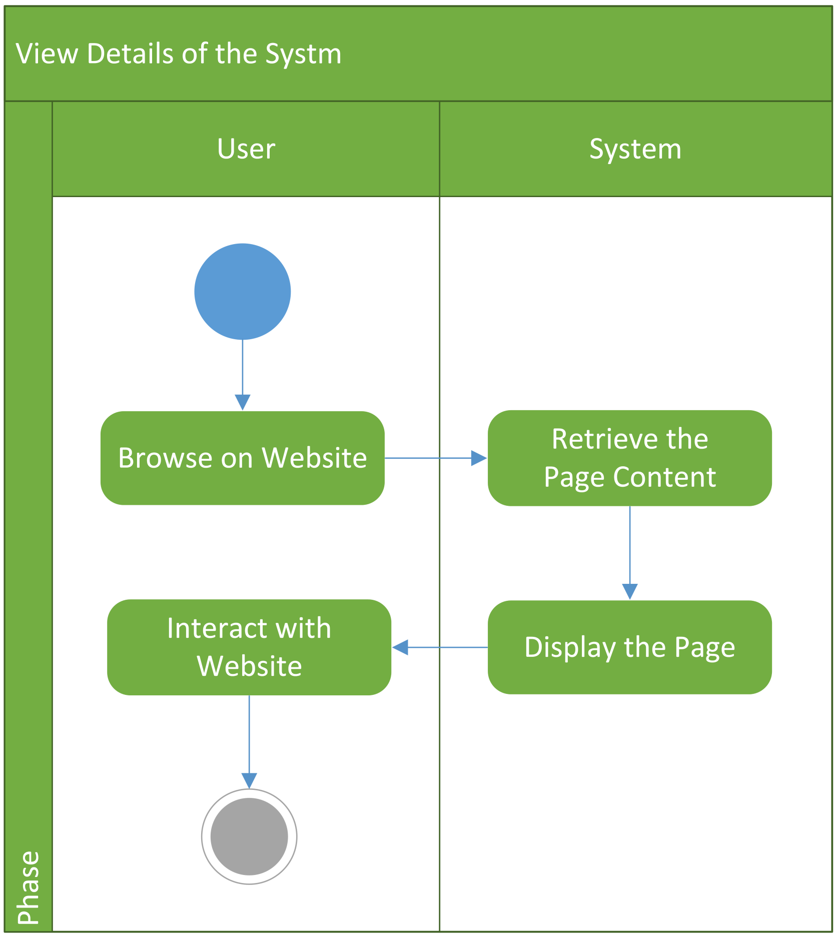


Figure 2.7: View Details of the System - Activity Diagram

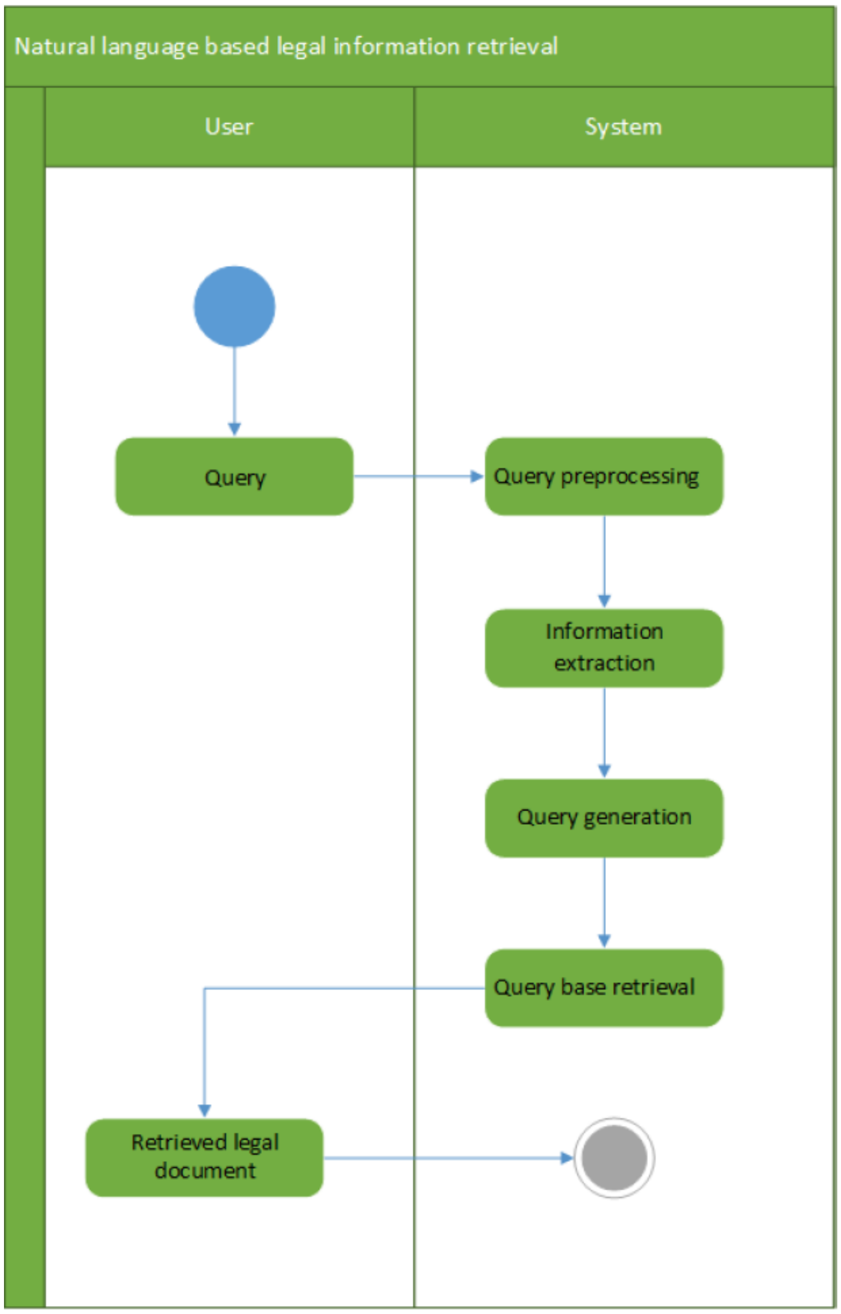


Figure 2.8: Query system using natural language text - Activity Diagram

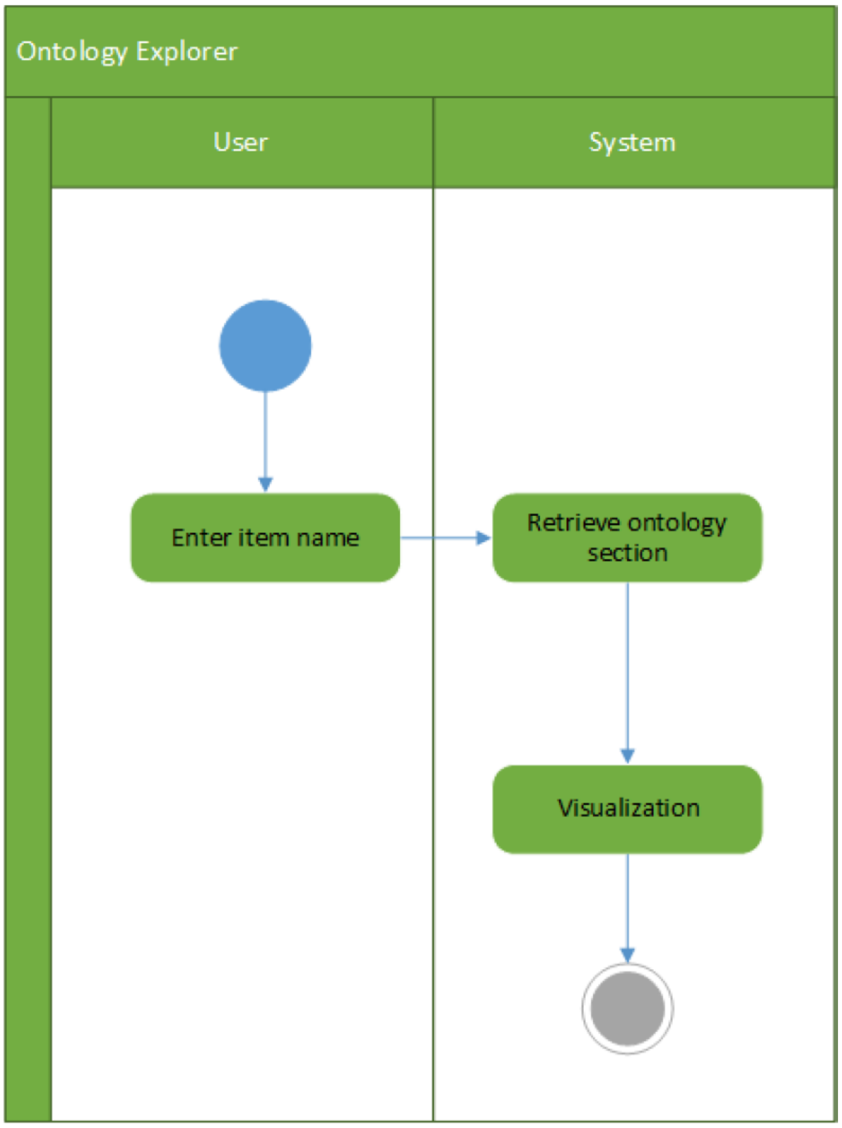


Figure 2.9: Explore the ontology - Activity Diagram

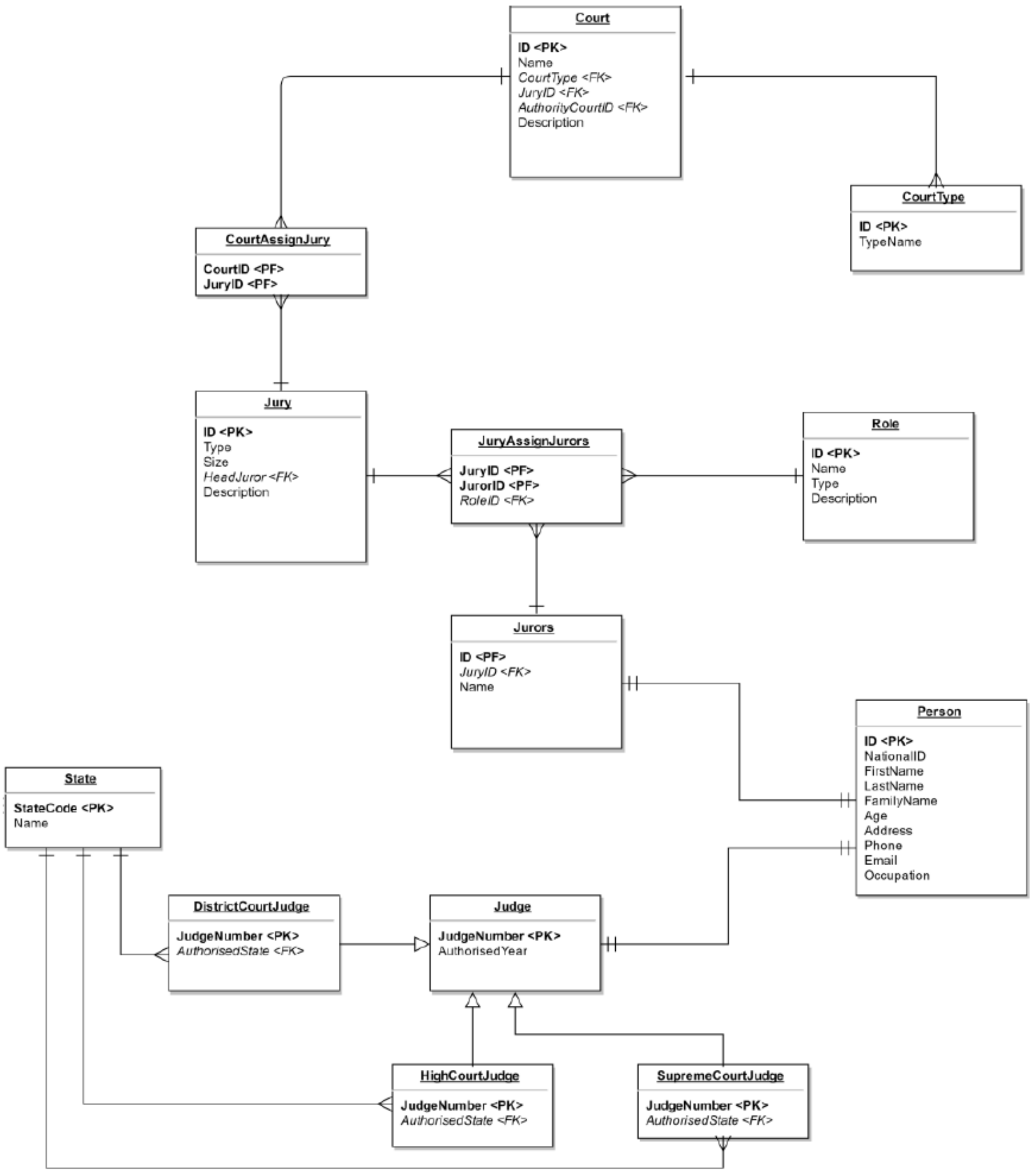


Figure 2.10: Data View Part 1

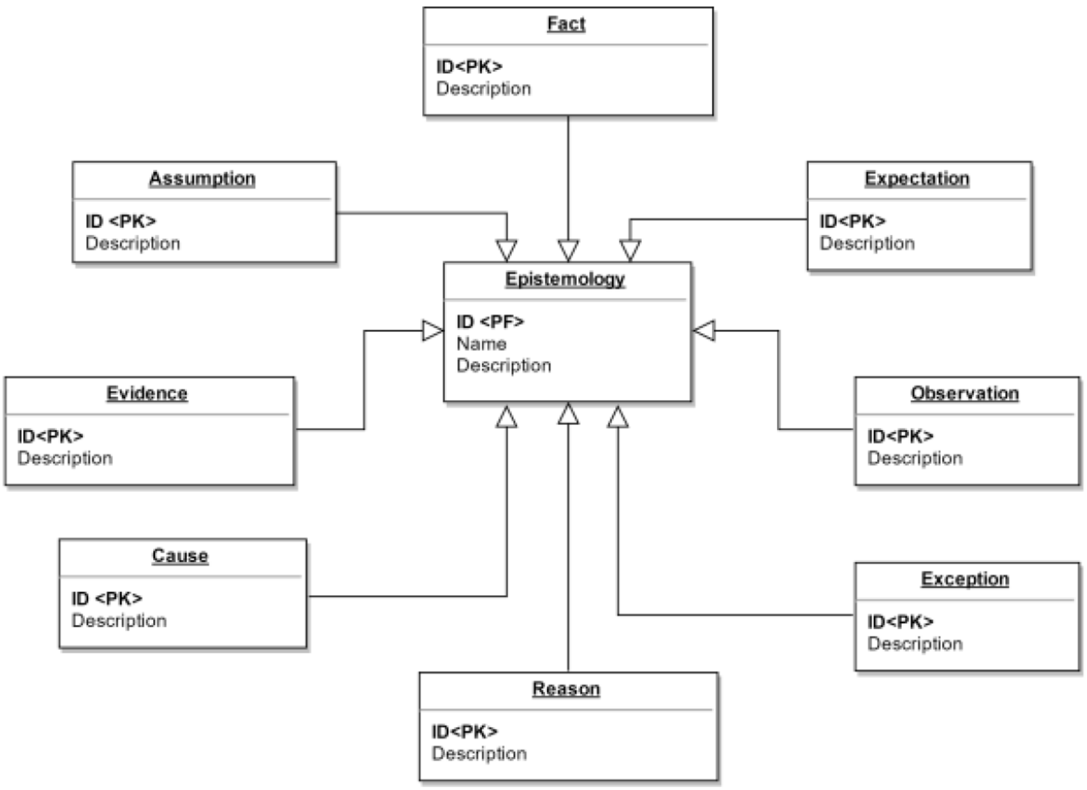


Figure 2.11: Data View Part 2

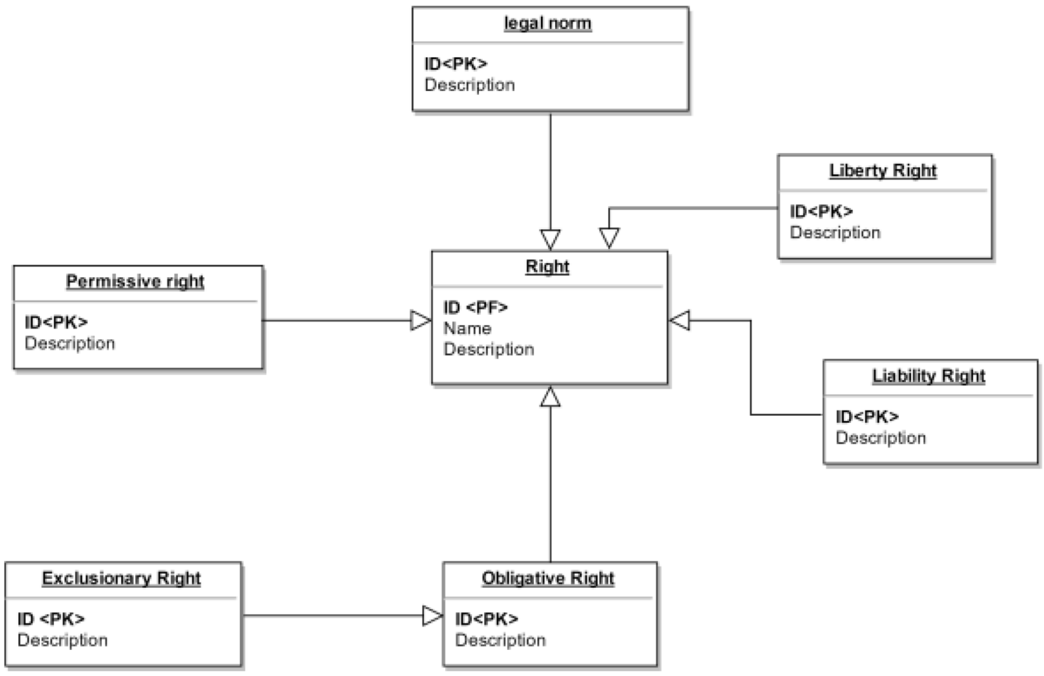


Figure 2.12: Data View Part 3

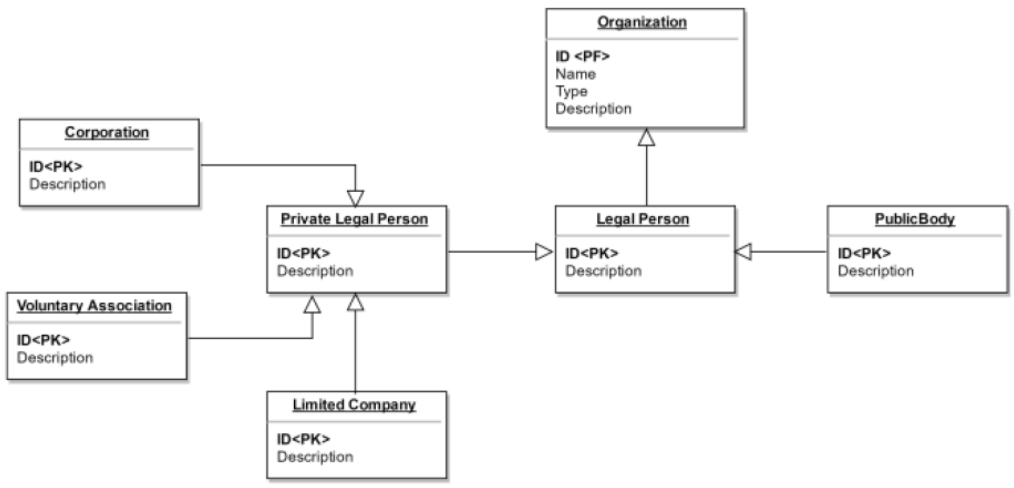


Figure 2.13: Data View Part 4

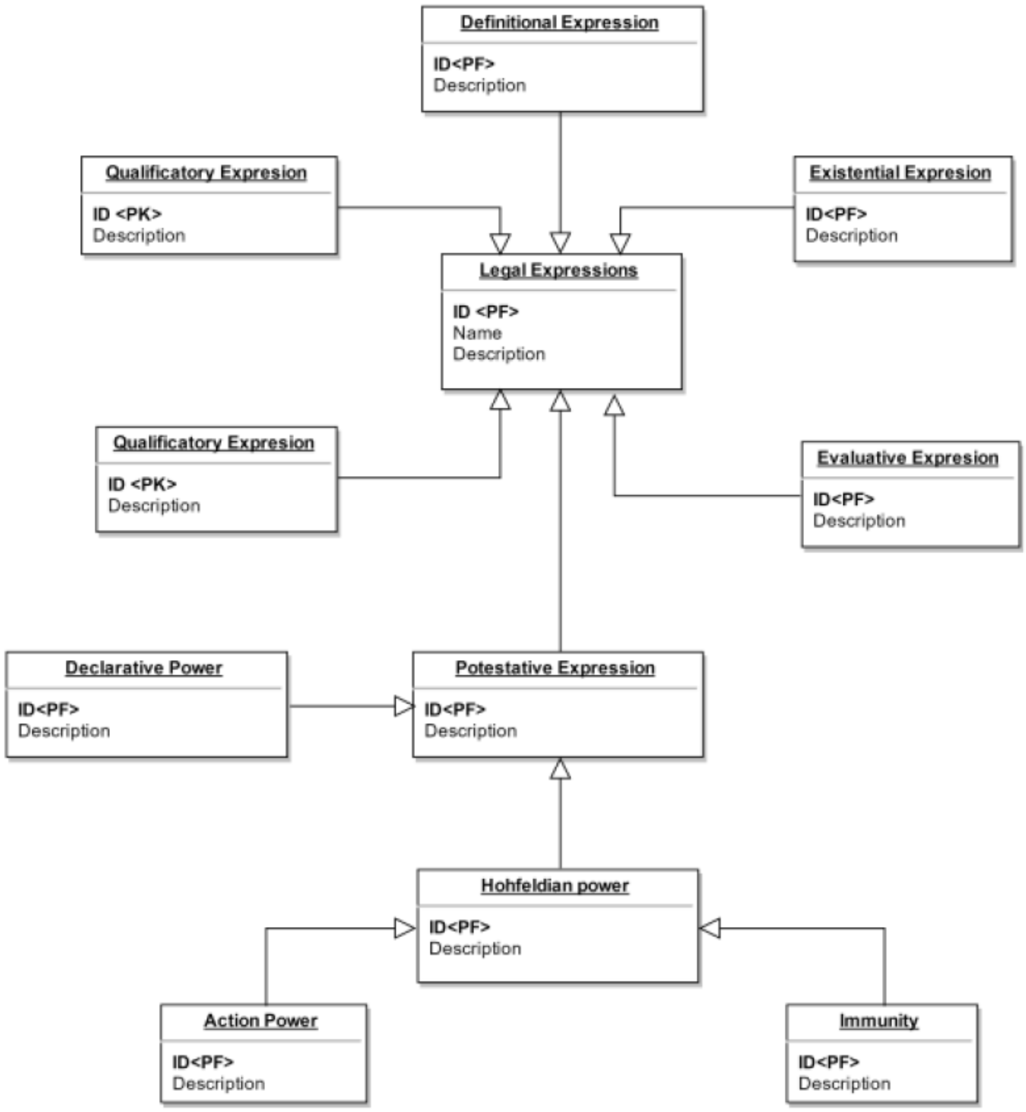


Figure 2.14: Data View Part 5

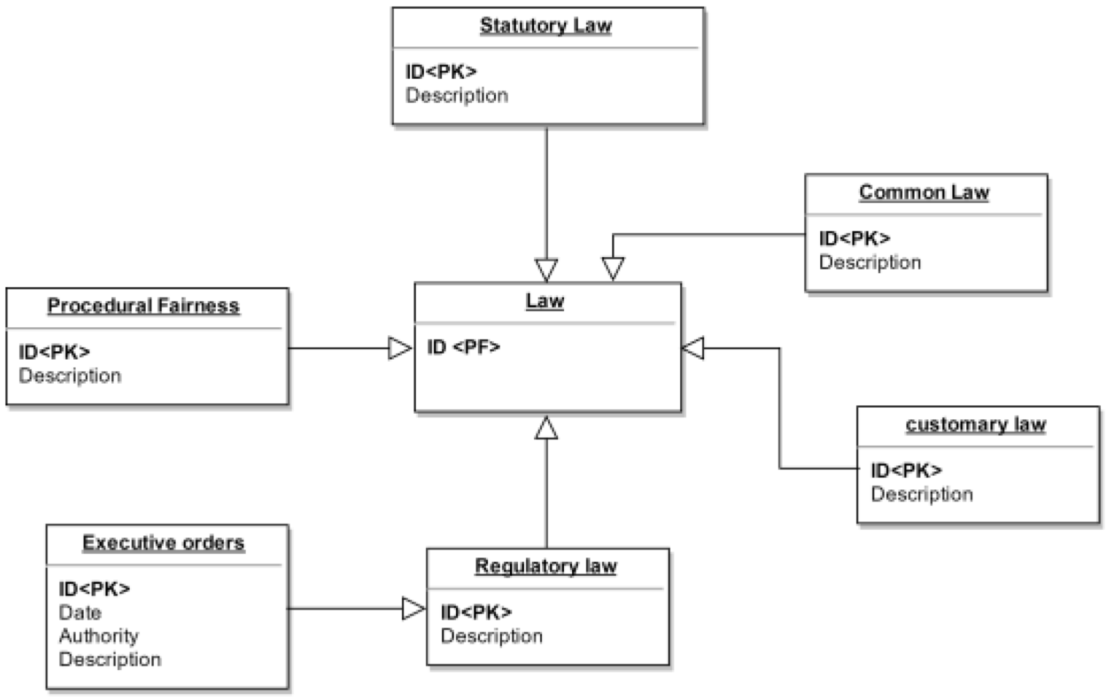


Figure 2.15: Data View Part 6

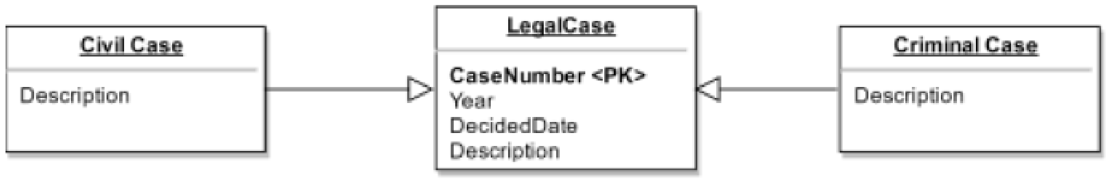


Figure 2.16: Data View Part 7

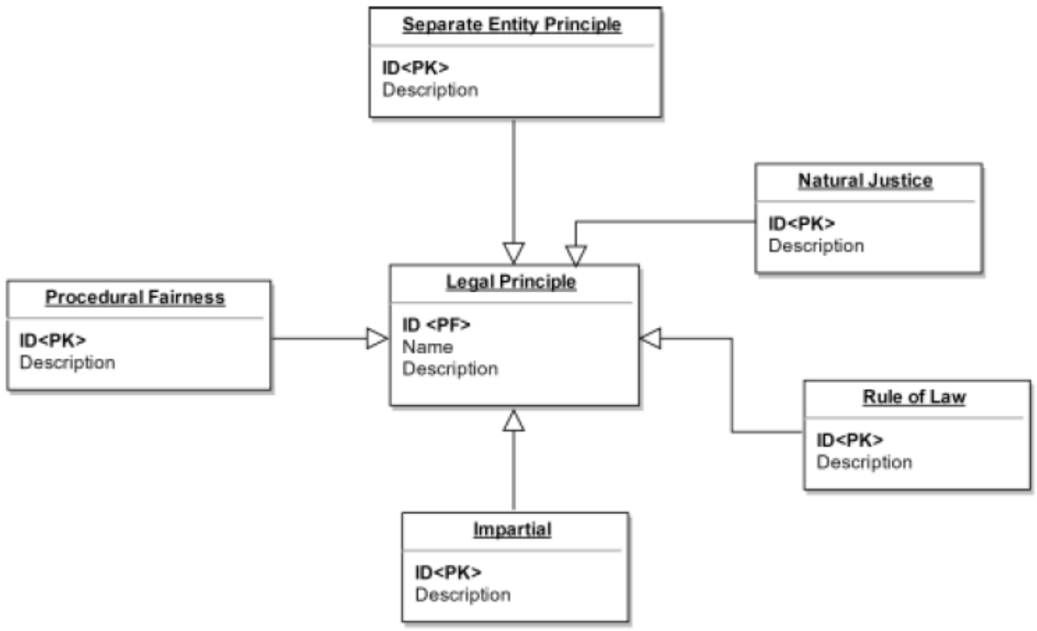


Figure 2.17: Data View Part 8

3 User Interface Design

This section describes the basic interfaces provided by the system to cater the major uses cases specified in Chapter 2. The purpose of this document is to provide a general idea about the view of the interface and how it will provide the required functionality to the interested user. Given below are the major user interfaces addressed in this section.

1. Landing Page of the Website
2. Services being Offered by the System
3. Pricing Structure of the Packages
4. The Team Members
5. Contact Us Page
6. Ontology Explorer
7. Natural Language Query Page
8. Legal Case Descriptions

3.1. Landing Page of the Website

The main preview of the website home page is depicted in Figure 3.1.

3.2. Services being Offered by the System

The services offered by this system is depicted in Figure 3.2

3.3. Pricing Structure of the Packages

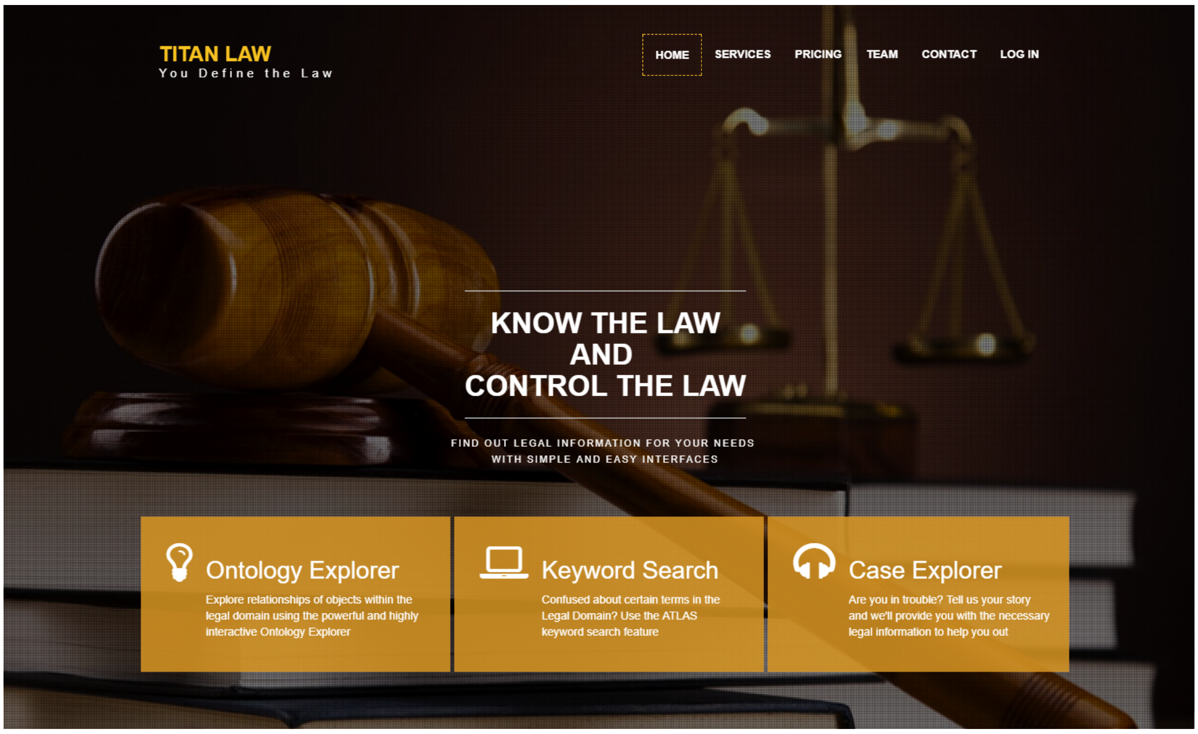
This section describes pricing structure and the packages available to the different kinds of users. This is depicted in Figure 3.3.

3.4. The Team Members

The might behind TitanLaw, is depicted in Figure 3.4, which includes the team members who developed this system.

3.5. Contact Us Page

This is the interview that allows the visitors to contact the team. This is depicted in Figure 3.5.


TITAN LAW
You Define the Law

HOME SERVICES PRICING TEAM CONTACT LOG IN

KNOW THE LAW
AND
CONTROL THE LAW

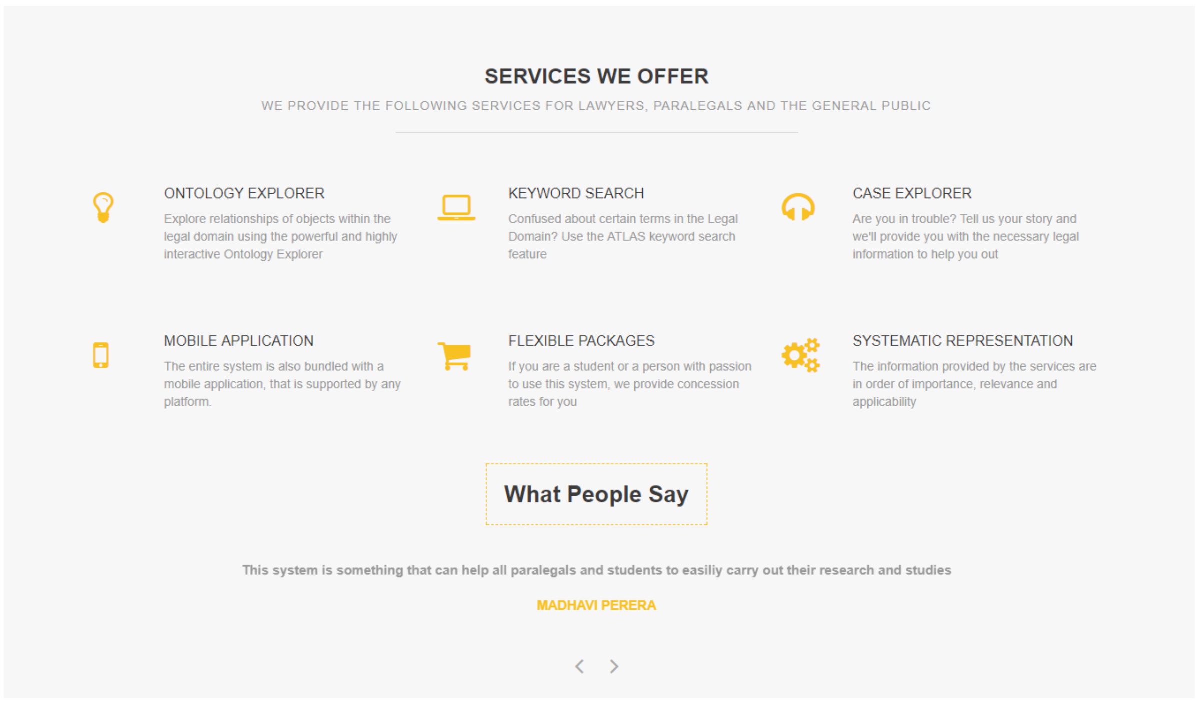
FIND OUT LEGAL INFORMATION FOR YOUR NEEDS
WITH SIMPLE AND EASY INTERFACES

Ontology Explorer
Explore relationships of objects within the legal domain using the powerful and highly interactive Ontology Explorer

Keyword Search
Confused about certain terms in the Legal Domain? Use the ATLAS keyword search feature

Case Explorer
Are you in trouble? Tell us your story and we'll provide you with the necessary legal information to help you out

Figure 3.1: Landing Page of the Website


SERVICES WE OFFER
WE PROVIDE THE FOLLOWING SERVICES FOR LAWYERS, PARALEGALS AND THE GENERAL PUBLIC

ONTOLOGY EXPLORER
Explore relationships of objects within the legal domain using the powerful and highly interactive Ontology Explorer

KEYWORD SEARCH
Confused about certain terms in the Legal Domain? Use the ATLAS keyword search feature

CASE EXPLORER
Are you in trouble? Tell us your story and we'll provide you with the necessary legal information to help you out

MOBILE APPLICATION
The entire system is also bundled with a mobile application, that is supported by any platform.

FLEXIBLE PACKAGES
If you are a student or a person with passion to use this system, we provide concession rates for you

SYSTEMATIC REPRESENTATION
The information provided by the services are in order of importance, relevance and applicability

What People Say

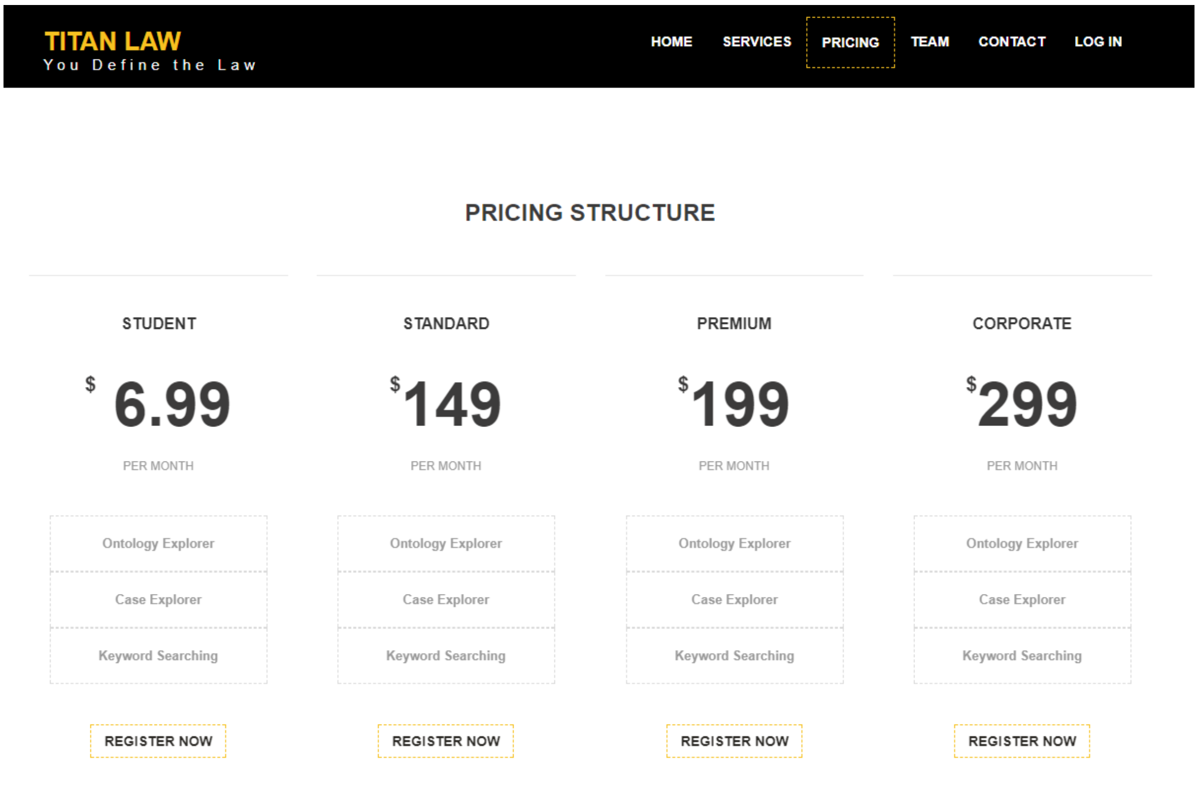
This system is something that can help all paralegals and students to easiliy carry out their research and studies

MADHAVI PERERA

Figure 3.2: Services being Offered by the System

3.6. Ontology Explorer

The interface shown in Figure 3.6 is the view of the legal domain ontology that allows the user to view the concepts and relationships within the legal domain.



STUDENT	STANDARD	PREMIUM	CORPORATE
$ 6.99	$149	$199	$299
PER MONTH	PER MONTH	PER MONTH	PER MONTH
Ontology Explorer	Ontology Explorer	Ontology Explorer	Ontology Explorer
Case Explorer	Case Explorer	Case Explorer	Case Explorer
Keyword Searching	Keyword Searching	Keyword Searching	Keyword Searching
REGISTER NOW	REGISTER NOW	REGISTER NOW	REGISTER NOW

Figure 3.3: Pricing Structure of the Packages

3.7. Natural Language Query Page

This is the interface that allows the user to query a user story, where the user needs better legal clarifications about. This interface shown in Figure 3.7, will return the relevant legal cases, related to the user's query.

3.8. Legal Case Descriptions

This interface depicted in Figure 3.8 shows the legal cases in terms of summaries as well as full legal case texts. With this the user has all the relevant legal cases at hand, in one single interface.

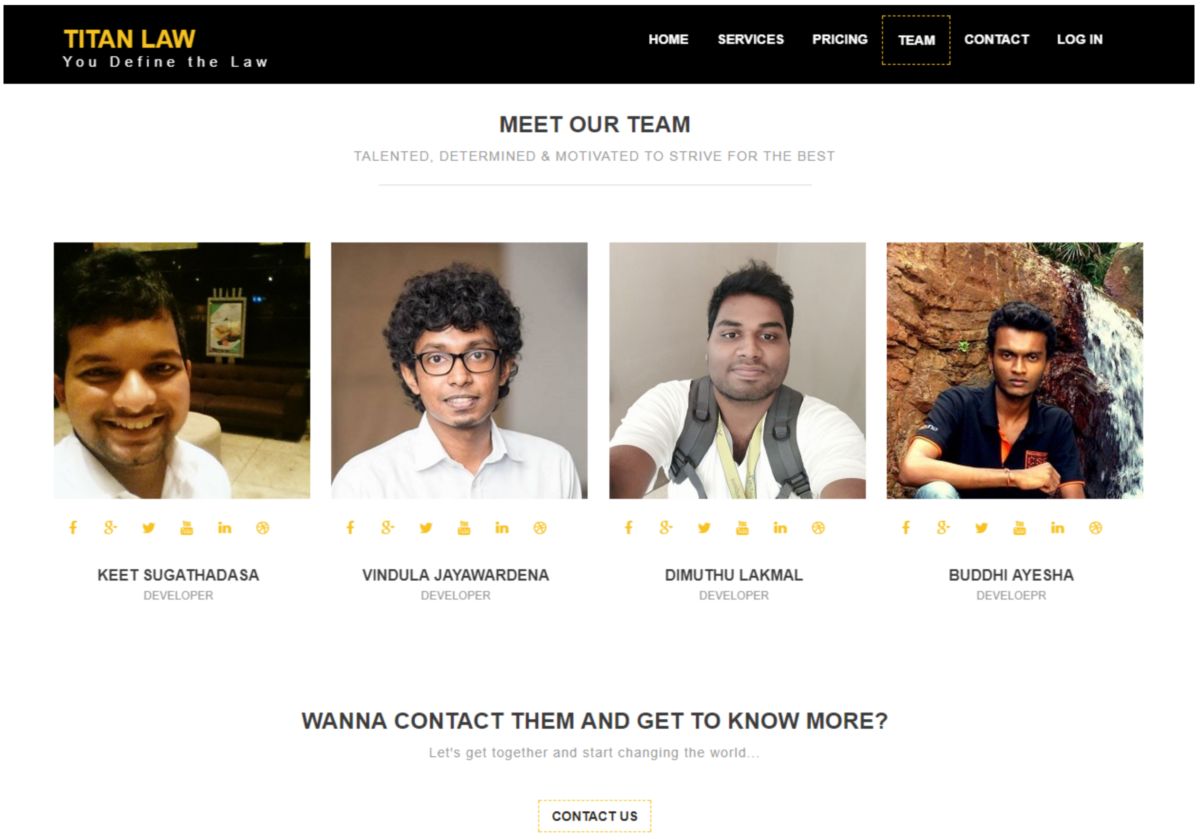
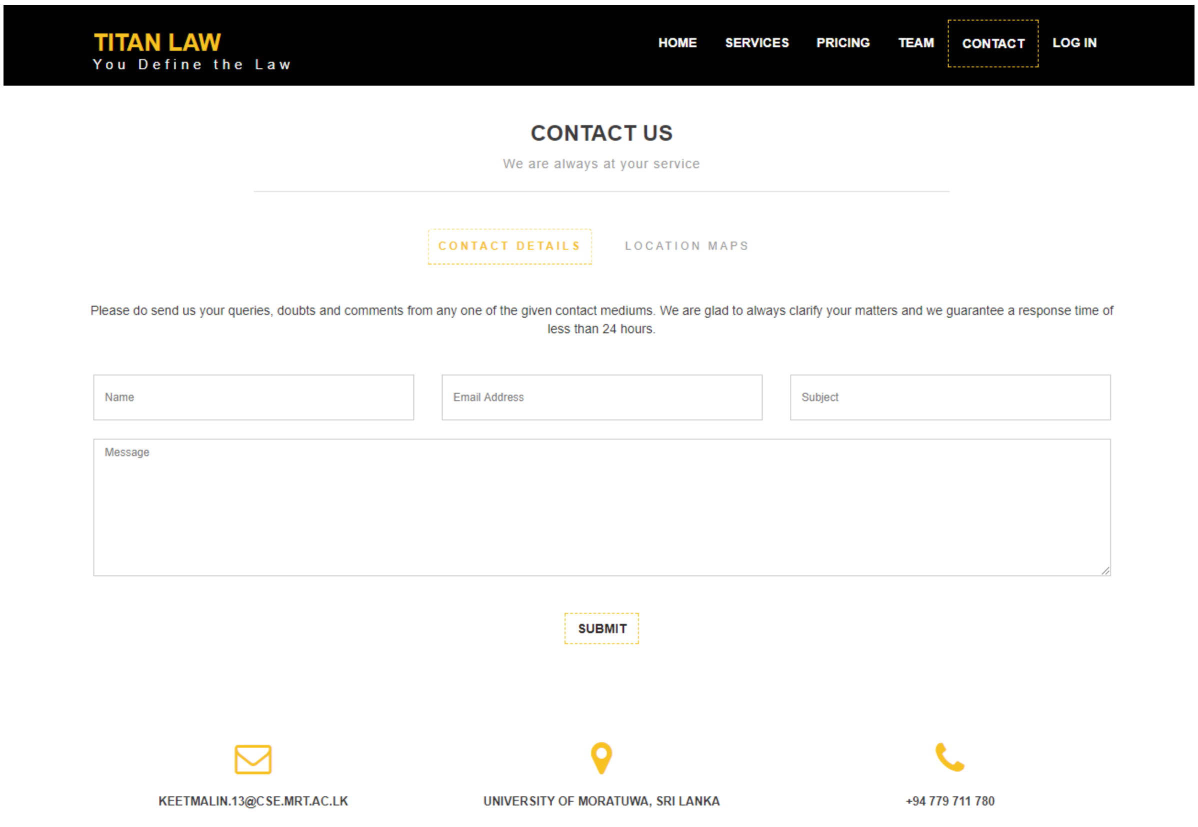


Figure 3.4: The Team Members



TITAN LAW
You Define the Law

HOME SERVICES PRICING TEAM CONTACT LOG IN

CONTACT US

We are always at your service

CONTACT DETAILS LOCATION MAPS

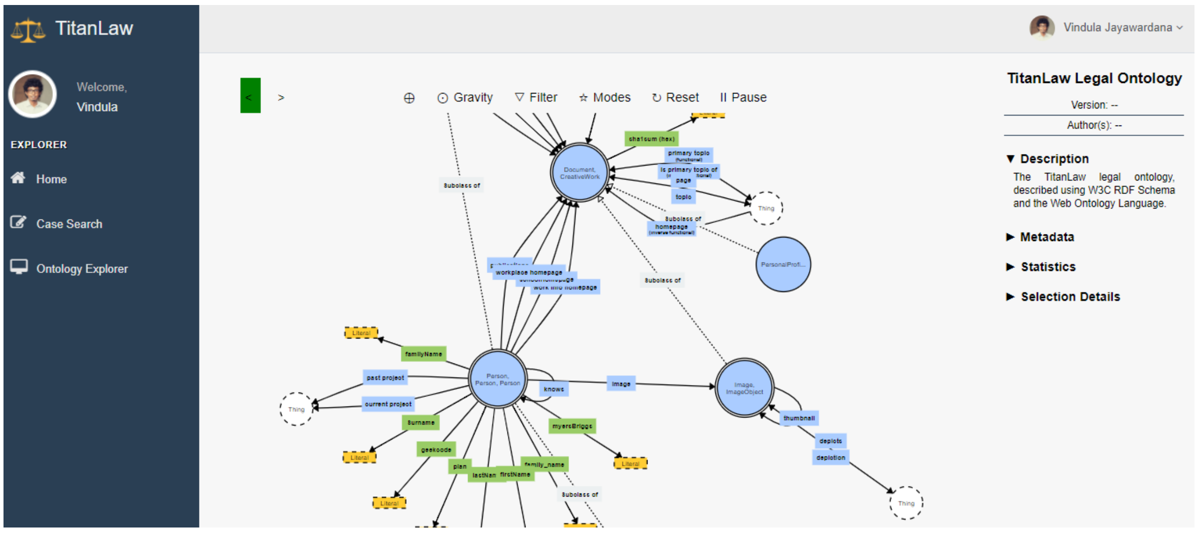
Please do send us your queries, doubts and comments from any one of the given contact mediums. We are glad to always clarify your matters and we guarantee a response time of less than 24 hours.

Name	Email Address	Subject
Message		

SUBMIT

KEETMALIN.13@CSE.MRT.AC.LK UNIVERSITY OF MORATUWA, SRI LANKA +94 779 711 780

Figure 3.5: Contact Us Page



TitanLaw Legal Ontology

Version: --

Author(s): --

▼ Description

The TitanLaw legal ontology, described using W3C RDF Schema and the Web Ontology Language.

▶ Metadata

▶ Statistics

▶ Selection Details

Figure 3.6: Ontology Explorer

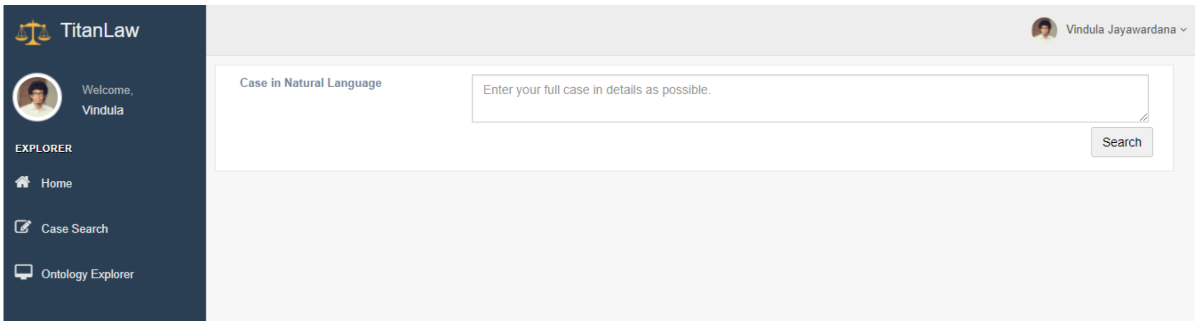
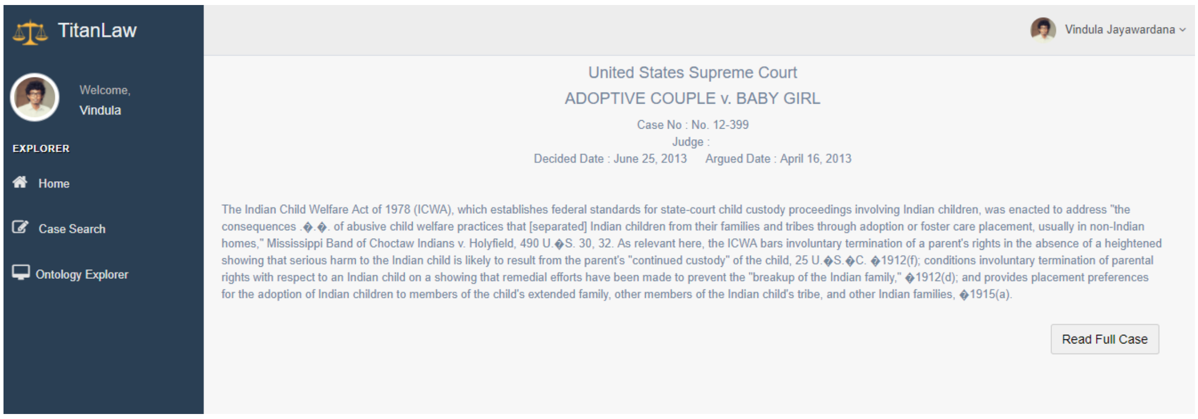


Figure 3.7: Natural Language Query Page



United States Supreme Court

ADOPTIVE COUPLE v. BABY GIRL

Case No : No. 12-399

Judge :

Decided Date : June 25, 2013 Argued Date : April 16, 2013

The Indian Child Welfare Act of 1978 (ICWA), which establishes federal standards for state-court child custody proceedings involving Indian children, was enacted to address "the consequences .�.�. of abusive child welfare practices that [separated] Indian children from their families and tribes through adoption or foster care placement, usually in non-Indian homes," Mississippi Band of Choctaw Indians v. Holyfield, 490 U.�S. 30, 32. As relevant here, the ICWA bars involuntary termination of a parent's rights in the absence of a heightened showing that serious harm to the Indian child is likely to result from the parent's "continued custody" of the child, 25 U.�S.�C. �1912(f); conditions involuntary termination of parental rights with respect to an Indian child on a showing that remedial efforts have been made to prevent the "breakup of the Indian family," �1912(d); and provides placement preferences for the adoption of Indian children to members of the child's extended family, other members of the Indian child's tribe, and other Indian families, �1915(a).

Figure 3.8: Legal Case Descriptions

4 Literature Review

Albeit the fact that ontology based information extraction has established its status in the research field [1], when it comes to the study of ontology based legal information extraction, there are only few attempts that have been made. In terms of the early attempts on the aforementioned research objective, J. L. Kolodner had performed a case based reasoning [37], while S. B. Bruninghaus and K. D. Ashley had addressed the need of improving the information representation from legal case texts with information extraction methods [4]. However, these efforts are limited by an approach based on text processing, without using linguistic information and important relationship descriptions available in the domain knowledge of the legal context. It is also important to identify systems which extract rulings from court opinions and retrieves relevant prior cases from a citatory database, based on natural language processing techniques and statistical methods [38], and systems whose approach incorporates linguistic aspects in their design [39]. However, the lack of domain knowledge incorporation of these researches, have created a research need for a more sophisticated system [5]

Addressing the above need in a simulated environment, Araujo et al [5] have exposed a novel way for achieving a more sophisticated system using an ontology, to drive the information extraction process. Albeit semantic information is widely used in legal domain ontologies and the fact that many legal documents are integrated with linguistic information, the system proposed by Araujo et al [5] has many practical limitations. The manual extraction process of terms from only ten court cases, creates a major practical issue in large legal domain corpuses. On the contrary, lack of proper preprocessing in terms of spelling checking and grammar checking, create certain doubts regarding the research as well. The heavy coupling and involvement of legal professionals in the linguistic step, which make the base for the remaining computational step, has also been a major drawback of their proposed system [5]. Due to the fact that this high coupling of legal professionals adds an extra constraint to the scalability of the system, the system proposed by Araujo et al [5] is expected to have a minimum coupling while it does not mean to fully alienate the legal professionals on the other hand. While being small in scale, the system proposed by Araujo et al [5] works as a proof of concept for the fact that OBIE can make an effective impact on legal information extraction, whereas scalable and extensive efforts are yet to be made.

However, there are some other approaches that have been taken to cater the same motive. A.Passerini et al [40] have used the 'Multi Class Support Vector' machine learning algorithm to do a classification based information extraction. But it can be clearly seen that, this system performs poorly compared to an ontology based information extraction system, according to given evaluation statistics [40].

Looking at the aforementioned efforts, it can be identified that the emphasis on a proper ontology based information extraction system, which is not entirely based on manual processing and which is not limited to simulated environments is an essential need. As Araujo et al [5] illustrate on the requirement of in-depth research on the same research area, what is proposed in this proposal is an extensive research with an eye towards eliminating the practical issues and hypotheses associated with the previous researches.

Loan Alfred LETIA et al present an ontological approach to the legal literature for translating sources of law into information accessible to people both with and without a legal education [3]. They also highlight the role of two legal ontologies; the LKIF-core Ontology and the Lex-is Ontology in improving the dissemination of legal knowledge. PEPIJN R. S. VISSER et al [2] provides a detailed comparison of four ontologies for the design of legal knowledge systems.

The rule based information extraction could be identified as a popular methodology for information extraction in various domains. The popularity of this methodology could be identified by the use of it in reliability critical medical domain [41] and as well as in the scientific domain [42]. While traditional approaches for rule based information extraction is primarily based on regular expression grammars, there are attempts on improving the scalability of such methods to large number of data sets and large number of rules such as Frederick Reiss at els ,algebraic approach to rule-based IE that addresses these scalability issues through query optimization [43]. However it should also be noted that as per Laura Chiticariu et al [44], while rule-based IE dominates the commercial world, it is widely regarded as dead-end technology by the academia. In the system developed by Walter and Pinkal [45], it describes a rule based information extraction technique which extracts definitions from legal court decisions. This system is being evaluated on a corpus of about 6000 German court decisions within the field of environmental law. By conducting a set of surveys on randomly selected verdicts, they have managed to identify some common structural elements found in legal definitions. This approach tries to use natural language processing techniques to create rules in order to extract relevant information from a given piece of text. Similar researches being carried out for this purpose are the systems proposed by Lame in 2005 [46] and Saias and Quaresma in 2005 [47], where both of these fall under information retrieval. In the system proposed by Walter and Pinkal [45], the general structure of the definitions of legal definitions are being modeled using the survey results. The limitation in this is that the survey has only covered a limited number of definitions where as the same definition can be represented in different structures in different online resources. Therefore the scope becomes limited to the definitions in interest and of the selected data source. This research has developed 33 such extraction rules, which gives a 50 hit rate and a precision of well above 70. The research carried out by Paulo and Teressa on identifying entities from judicial documents using linguistic information and machine learning techniques [48] which uses support vector machines for this purpose.

In the case of machine learning, if the output space has no structure except whether two elements are equal or not, we have a classification task. The purpose of text classification is to classify each input object into a pre defined class in the system. Each element of the output space is called a class. The supervised classification task of natural language texts is known as text classification [48]. This text classification task has being a very dominant research area in the machine learning field. Algorithms such as decision trees [49],linear discriminant analysis [50], naive Bayesian algorithms [51] and support vector machines [52]. The reason for Paulo and Teressa [48] to go forth with the use of Support vector machines for text classification is that in [52], it says that using SVMs to learn text classifiers is the first approach that is computationally efficient and performs well and robustly in practice. There is also a justified learning theory that describes its mechanics with respect to text classification. The methodology, uses a support vector machine classifier to associate concepts to legal documents and a natural language parser to identify named entities, namely, locations, organizations, dates, and references to other articles and documents.

For identification of Legal Concepts, Paulo and Teressa [48] has used the bag of words technique [53] in order to carry out the preprocessing of the textual representations, and it is being run on SVM light [54] with a linear kernel and other default parameters. As the first step, this approach uses the Bag of Words technique to represent each document being used. Afterwards it uses the Stratified Cross Validation technique [55], which is a model evaluation method that divides the original dataset into subsets, each with the same distribution of examples, between categories as the original dataset [48]. Then one of the subsets is used and the remaining subsets are used to form a training set. This procedure is repeated for each subset separately and the variance of the resulting estimate is reduced as the number of subsets increase. But, the system proposed by Paulo and Teressa [48] has The problem that this system shows good results in terms of precision but does not show much good results in terms of recall. In terms of named entity recognition, the referencing does not have a number of unique references and it requires further text processing of the documents. According to the evaluation, the concept identification task has a better precision for English, German, Italian and Portuguese, whereas worst results were obtained for Romanic languages. The named entity task it has shown good results for date identification, but bad results for identification of organizations and references to other articles and legislation.

4.1. Information Retrieval

Information retrieval is finding material of an unstructured nature that satisfies an information need from within large collections of documents [35]. The task of document retrieval has far more reach into research areas [56] such as video/song identification [57], newspaper categorization and retrieval [58], and multilingual document retrieval systems. Most of these systems commonly use skip-grams, Bag of Words (BOW), or term frequency inverse document frequency (TF-IDF) [59]. Day by day, the field of information retrieval get optimized to provide better results for users' information needs.

The modest way of identifying similarities between documents is by measuring the distance between each of the documents in a given vector space. But the problem arises with the way we represent documents in order to get a more accurate and precise representation for each of the documents. As mentioned in this study, previous studies have also tried to address this problem with various combinations of weighting mechanisms. Salton has addressed this issue by trying different combinations of statistic measures and term frequencies [56], whereas Perina has come up with a Componential Counting Grid for learning document representations [60]. TF-IDF [59] is a text mining technique, that gives a numeric statistic as to how important a word is to a document, in a collection or a corpus. This relative importance of words in a document is used to retrieve and categorize documents in relation to one or more query words.

4.2. Information Extraction

4.2.1. Introduction

The activity of processing human language text by means of Natural Language Processing (NLP) is an open research area, that incorporates many computational techniques for different fields of texts. In the field of Natural Language Processing (NLP), text mining is the process of automatically or manually analyzing a given piece of natural text and deriving high quality information from text. This process is also known as knowledge discovery from textual databases which refers to the process of extracting interesting and non-trivial patterns or knowledge from text documents [61]. Text mining is a multi-disciplinary field that involves, information retrieval, text analysis, information extraction, clustering, categorization, visualization, database technology, machine learning, and data mining. Information extraction is one of the major subfields under text data mining, where the general idea is to automatically retrieve certain types of information from natural language text [1].

The field of information extraction started in the 1970s, which was also the early days of Natural Language Processing. The need for information extraction came up with the financial traders of that day, where they needed to implement a system which is able to provide real time financial news. As the first commercially available information extraction system, the Carnegie Group for Reuters Limited, developed JASPER, which is a template driven approach with partial understanding techniques, and heuristic procedures to extract certain key pieces of information from a limited range of text, to cater to the need of real time financial news [62].

Addressing the task of information extraction for different domains, comes with both syntactic and semantic difficulties that make the entire process cumbersome. Even though a general architecture can be defined for information extraction tasks, as in Section 4.2.2.A, domain specificity disrupts the entire process of information extraction due to its complexities in both semantics and syntax. Many researchers address this problem via domain models, extraction patterns, and extraction rules, which require a heavy coupling with domain experts [5]. Most of the time, it requires a domain expert annotated model to execute the process of information extraction. But, there have been studies that uses information formats, which are not defined a priori by experts in a field, rather given a set of texts in a sub-language domain, the information formats are induced by using distributional analysis to discover word classes in the domain [63].

Information Extraction was spurred by a series of Message Understanding Conferences (MUCs). When it comes to MUCs, domain specificity plays a key role, where the need for domain specific information extraction systems have risen to bring the MUC forward as a competition based conference [64]. In chronological order, naval operations messages, terrorism in Latin American countries, joint ventures and micro-electronic domains, news articles on management changes, and satellite launch reports, are a few of the domain specific information extraction systems which were addressed at the Message Understanding Conferences in the 1980s and 1990s. With this, many other fields like medicine, geology, biology and legal domains emerged themselves into the field of domain specific information extraction.

In this study, our interest is mainly focusing on the legal domain, where the known approaches and trends used for the process of information extraction is discussed. These approaches and trends may be limited to different domains, whereas incorporation of such methodologies can be used with variations, to support other domains as well. In consideration of the legal domain, there is a massive legal literature in printed documents as well as on-line, where the current accessibility of sources of law does not suffice to serve legal and non-legal professionals [3]. Many researches have approached this problem with different mechanisms to provide efficient and an accurate legal information extraction systems to cater for this purpose. Within the fields of artificial intelligence and law, it has hardly ever been tried to make the contents of sources of law, and the relations among them, more accessible to those without a legal education [65].

The difficulties of forming a proper legal information extraction system comes in many different aspects, where majority of the issues are related to the correct identification and representation of legal information. As Araujo et al [5] specify, most of the well known approaches lack the use of important relationship descriptions available in the domain knowledge of the legal context and linguistic information. Despite the difficulties in the process of information extraction from legal domain, researchers have addressed the need with different approaches in mind. On an overview, Kolodner had performed a case based reasoning [37], while Bruninghaus and Ashley had addressed the need of improving the information representation from legal case texts with information extraction methods [4]. It is also important to identify systems which extract rulings from court opinions and retrieve relevant prior cases from a citatory database, based on natural language processing techniques and statistical methods [38], and systems whose approach incorporates linguistic aspects in their design [39]. Araujo et al [5] have exposed a novel way for achieving a more sophisticated system using an ontology, to drive the information extraction process. Passerini et al [40] have used the ‘Multi Class Support Vector’ machine learning algorithm to do a classification based information extraction.

In recent history, there have not been any thorough reviews on information extraction techniques within the legal domain, which points out the need for a comprehensive and up-to-date survey study, specifically for the legal domain. Araujo et al. has done a survey on current approaches and trends in the legal domain for both information extraction (IE) and information retrieval (IR) [66]. However, that study is very limited in scope and it has provided both IR and IE, which in turn reasons out the limitations on finding current known approaches, available in general for the legal domain. Rather, there are some other ongoing studies related to information extraction in the legal domain, which are guided by various domain specific models [5].

This study presents a survey of available legal information extraction systems in terms of the methodologies being used, scope, and limitations; given that most of these approaches tend to be successful only for some specific criteria with certain restrictions on them. For example, the system addressed by Kolodner [37] as well as the system addressed by Bruninghaus and Ashley [4], are limited by an approach based on text processing, without using linguistic information and important relationship descriptions available in the domain knowledge of the legal context. The ontology based legal information extraction system proposed by Araujo et al [5] has many practical limitations. The manual extraction process of terms from only ten court cases, creates a major practical issue in the large legal domain corpora which has hundreds of thousands of court cases and other relevant documents.

4.2.2. Defining Legal Information Extraction Systems

In preparation for the definition of legal information extraction systems, the general idea behind information extraction (IE) need to be understood. In succinctly, Wimalsoorriya et al. state, information extraction (IE) as automatically retrieving certain types of information from natural language text [1]. As defined by Russell and Norvig [67], IE aims to process natural language text and to retrieve occurrences of a particular class of objects or events and occurrences of relationships among them. Adhering to the same idea, Riloff presents a similar definition stating that IE is a form of natural language processing in which certain types of information must be recognized and extracted from text [68].

As an example IE system in legal domain, we can illustrate a system that extract information pertaining to a court, judgment or claim. Given a set of legal cases, the system should be containing a model to guide the process on what to look for out of the natural text corpus. This model retrieves the information matching the stated criteria and ignore the rest of data.

On more descriptive basis, Russel and Norvig [67] stated that information extraction lies in between information retrieval systems and text understanding systems. Being a system which finds documents that are pertaining to a user need, information retrieval systems have been a major adoption over the past few decades. On the contrary, text understanding systems attempt to analyze text and extract their semantic contents [69]. However, Wimalasooriya at al. devise the fact that since the IE systems lies between the IR and text understanding system, their success has also been somewhere in between the levels achieved by IR and text understanding systems. On the other hand, the robust, efficient and high-coverage shallow text processing techniques which have emerged with recent advancements in the field of Natural Language Processing (NLP) have contributed to the spread of deployment of IE techniques in real-world applications for processing of vast amount of textual data [70]. Furthermore IE has compose a foundation technology in many other NLP applications, such as Machine Translation, Question Answering, Text Summarization, Opinion Mining, etc.

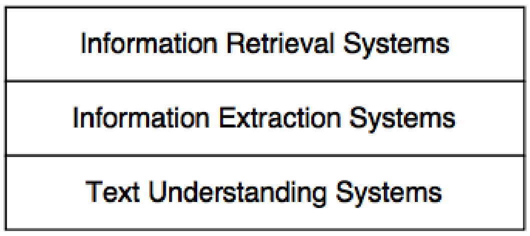

Information Retrieval Systems
Information Extraction Systems
Text Understanding Systems

Figure 4.1: Three layers of text processing systems

In this study, we have attempted to arrive at a definition for a legal information extraction systems with the focus on the key characteristics of information extraction systems and by identifying the key factors which differentiate legal domain from the others. These are presented below.

- Process unstructured and/or semi-structured machine readable natural text : IE is most prominently considered as a subfield of natural language processing which deals with human language text [71]. These text can be either unstructured or semi-structured. Unstructured text like text files, do not attempt to preserve the structure awareness while semi-structured text like web pages adhering to a format, attempt to preserve the structure awareness to a some extent. These text on the other hand should be machine readable to facilitate automatic processing [72].
- Automatically extract structured information : Applying information extraction on natural text focuses at creating a structured representation of information [72] or in other terms, a representation of the information that is machine understandable. Looking at the typical IE tasks such as Name Entity Recognition (NER) which addresses the problem of identifying pre-defined named entities, Co-reference Resolution (CO) to identity multiple mentions of same entity in text, Relation Extraction (RE) to classify relationships between entities, and Event Extraction (EE) to identify events out of natural text, the need for a structured representation of information could be identified [70].
- A domain expert annotated model to guide the information extraction process : As mentioned previously in this study, an IE system should be containing a model to guide the process on what to look for out of the natural text corpus. This model may different from one system to other, however it should be capable of retrieving the matching information for the stated criteria. Most of the time, the model to guide the IE process is a domain expert annotated model. But however, it is not a necessary condition to be satisfied. There are studies that uses information formats, which are not predefined a priori by experts in a field, rather, given a set of texts in a sub-language domain, the information formats are induced by using distributional analysis to discover word classes in the domain [63]. The guiding model can be developed by using many NLP techniques as explained in the later part of this study. As an overview, ontology driven models [1], rule driven models [41] and template driven models [73]can be identified as prominent annotated models on catering the expected purpose.

Apart from the key characteristics of IE system as mentioned above, following key factors in relation to legal domain is also need to be considered in devising a general definition for legal information extraction systems.

- A multilingual text corpora : The nature of legal text corporas generally includes multilingual contexts [74]. For an example, a text corpus written in English may have some content written in Latin. This make the IE process to be complicated and error prone.
- Based on semi-structured information repositories : Most of the repositories available for legal domain text corpora are semi-structured by the nature [75]. For an example a legal case may contains hyperlinks to many other relevant cases of interest, within the case itself. Hence this inherent semi-structured nature need to be handled accordingly in an IE process.
- Domain specific semantics and usage : Legal domain has an inherent property of being semantically difficult to elaborate. Being a profession which deals with logical inferencing, many domain specific

semantics and usage of them involves in the processing [74]. This applies to the legal text corpora as well. Hence, in terms of domain specific IE, a special effort may need to be put in legal domain related information extraction.

Combining these legal domain specific key factors with aforementioned information extraction characteristics, we provide the following definition for the legal information extraction systems.

Legal Information Extraction System (LIES) : A system which perform the task of automatic extraction of structured information from unstructured and/or semi-structured machine-readable and semantically domain coupled documents using an annotated model for guiding the information extraction process.

A. General Architectures of Legal Information Extraction Systems

We do not attempt to redesign the general architecture of legal information extraction systems given the reasonable fact that all information extraction systems shares the same underlying architecture if we are to consider the high-level overview. However as the IE process is highly coupled to the domain of interest, the low level architectural overview may have slight differences. Therefore, here we present a high level overview of the general architecture of an information extraction system as defined by Jakub Piskorski et al. in [70] as illustrated by figure 4.2.

As illustrated in figure 4.2, a typical IE system would composed with a domain-independent linguistic analysis and domain specific core IE component.

On a more descriptive note, within the domain independent linguistic analysis component, included sub components are performing the following sub tasks.

Metadata Analysis - This performs the extraction of document specific characteristics. For an example, it captures the language information which the document is written of. In addition, it identifies the structure of the documents by identifying title, body and paragraphs.

Tokernization - This component performs the segmentation of the text to units called tokens. Essentially a word is a token. Besides that, it also identify the type of the tokens such as identification of capitalized words, words written in lowercase letters, hyphenated words, punctuation signs, numbers, etc.

Morphologic Analysis - Upon tokenizing, morphologic analysis performs the extraction of morphologic information such as lemma (base form) or part-of-speech from tokens. In addition to that, other morphological tags depending on the part-of-speech such as tense, mood, aspect and person kind of features of verbs will also be extracted. Typically part-of-speech disambiguation is carried out to resolve ambiguousness of words with respect to certain morphological information category.

Sentence Boundry Detection - This outlooks the text as a sequence of sentences. Upon identification of such sequence of lexical items together with their features, a segmentation will be performed to extract that sequence.

Common Named Entity Recognition - This performs the typical NER task. However, this identifies only the named entities such as numbers and currency, which in fact are domain independent named entities.

Phase Recognition - Noun phrases, verb groups, prepositional phrases, acronyms, and abbreviations will be recognized at this step of the domain independent linguistic analysis.

Syntactic Analysis -As the final step with the main component for domain independent linguistic analysis, this computes the dependency structure or parse tree from the sequence of lexical items and small-scale structures identified within a sentence.

Upon performing the domain independent processing, next up is the domain specific linguistic analysis which performs the core IE tasks such as NER, co-reference resolution, and detection of relations and events. Since the aforementioned core IE tasks are typically domain dependent, it is essential to cope with them at the domain specific linguistic analysis component. It should be noted that, the extent to which domain specific analysis and domain independent analysis are performed, may vary on the type of application or the domain of interest.

In order to cater for domain specific semantic and inherent usage of it, within the domain specific IE component, a yet another named entity recognition is applied to identify the entities relevant to a given domain. It is important to note that, in practice, the borders between domain-independent linguistic analysis components and core IE components may be blurred. For an example, there may be a single NER component which performs domain independent and domain specific named entity extraction simultaneously.

Pattern matching is essential for two major reasons. 1. To identify text fragments, which describe the target relations and events. 2. Extract the key attributes to fill the slots in the template representing the relation/event.

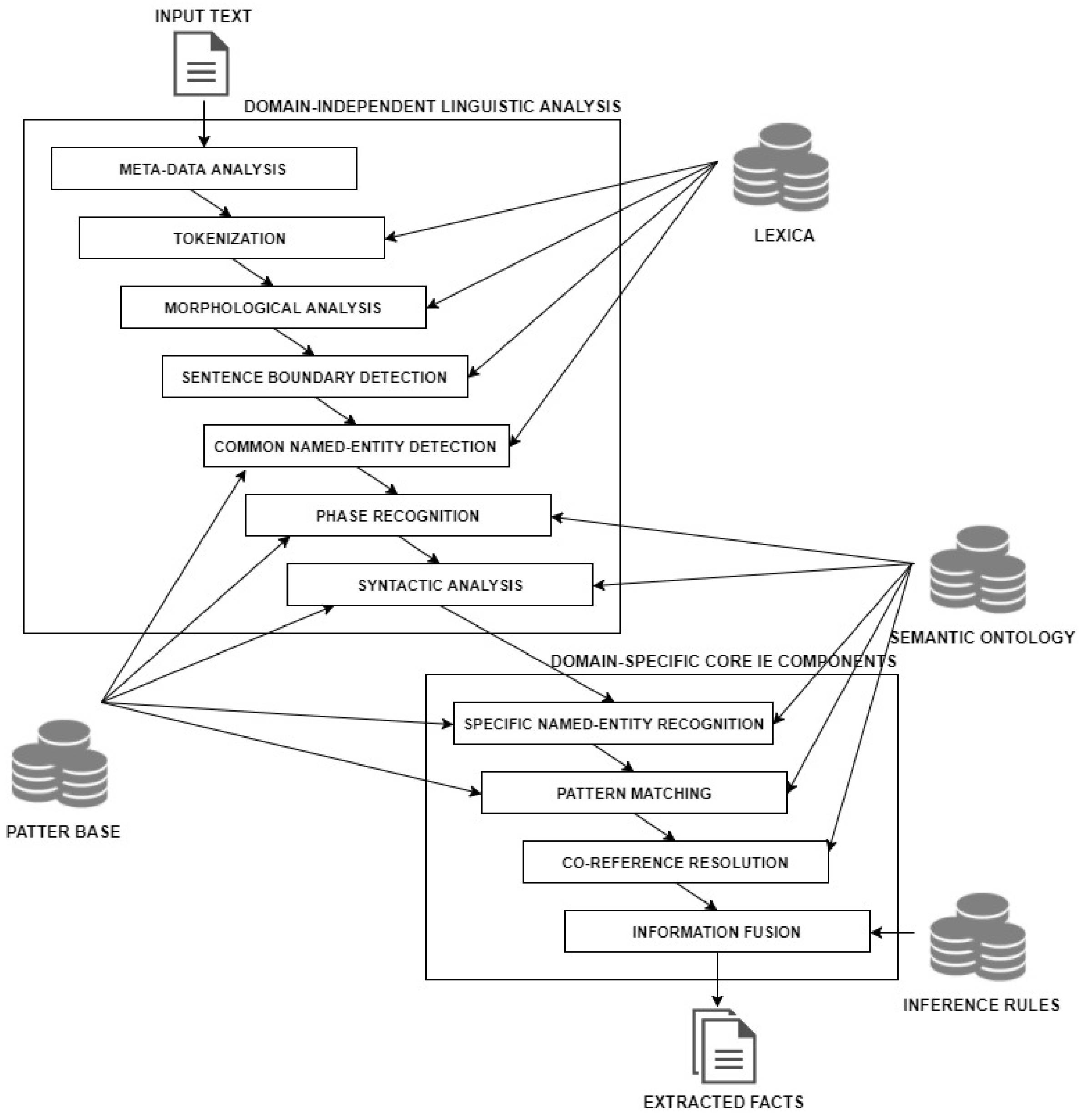


Figure 4.2: General Architecture of an Information Extraction System

A co-reference resolves the task of finding all expressions that refer to the same entity in a text. For an example, given the sentence *The judge Salmon warned Ben as he intentionally tried to keep the silence*, this component would identify, both *Ben* and *he* refer to the same entity within the context.

As the final step of the process, partially-filled templates are fused and validated using domain-specific inference rules in order to create full-fledged relation/event descriptions. Since the relevant information may be scattered over different sentences or even documents, it is essential to perform the information fusion at this stage to rectify that.

4.2.3. Legal Domain and Information Extraction Systems

The main goals of information systems is to collect information and organize them in a useful manner, and put information in a semantically precise form that allows further inferences to be made by computer algorithms [76]. With the growing set of documents and on-line resources, it is made important to implement information extraction (IE) techniques within systems, in order to extract relevant and necessary information. These IE techniques can be made fully automatic or semi automatic, depending on the task at hand and the domain of interest.

As defined in Section 4.2.2, legal information extraction systems do need to consider the fact of having semi-structured and domain specific structural corpora, when implementing the extraction process. In most cases, there is always a certain structure being followed in legal documents, which are specific to the domain. The most important ones are the legal institutional structure, then the related juridical processes, followed by the legal documents internal structure and, finally, the contextual use of concepts and sentence constructions [66]. These are mainly structural problems that need to be addressed when implementing the information extraction process.

Similarly, problems arise in the practical context of legal domain and its usages. These issues primarily validates the need for legal domain specific information extraction. When a lawyer or a paralegal is faced with a court case or a similar task, he or she will have to go through a lot of documents by researching on the relevant case to get the necessary information needed for that case. This information may consist of different aspects, such as previous court cases, judgments and laws which will have a major impact on the given case. However, due to the way that this information has been stored, legal officers find it extensively painful to carry out manual searching, for the relevant information of a given case. Apart from that, this manual process is subjected to erroneous situations where a small human error could make a huge impact on a given case.

Legal experts perform relatively difficult legal clerical work that requires accuracy and speed. These legal experts often summarize legal documents, such as court judgements, and look for information relevant to specific cases in these summaries [77]. So finally, whenever a legal officer want to find relevant legal information, he or she has to understand, interpret, explain and research a wide variety of documents in the legal domain.

To avoid those detrimental scenarios, online systems and frameworks like WestLaw [78] and LexisNexis [79] came into play. However, due to the fact that the aforesaid systems only provide query based searching, where legal officers need to remember keywords which are predefined when querying for relevant legal information, there is still a hassle in accessing this information. There is no simple and easy way for lawyers to enter some natural text on a given case and get the information relevant to that case effectively and reliably. This raises a potential need for a proper system to do hassle free legal information extraction based on natural text queries.There have been some attempts built on rule based, case based and statistics based methods, to overcome the aforesaid requirement.

It should also be noted that access to legal information and legal literature is a fundamental democratic right [67] of all citizens of a country irrespective of the fact that they have had a legal education or not. Having that kind of a legal background, arises the need for a proper system which allows any citizen to access the legal information, irrespective of his or her education level. When a normal citizen requires help in the legal sector, that person will reach a legal officer to get aid. But the problem they face is that, manual summarization of legal judgments is relatively more expensive due to the involvement of legal experts in the process. But this is also subject to misinterpretations and erroneous summaries. [77] There are currently systems like the work of Farzindar [80] , available in the legal domain for automatic production of legal text summarization, but they have their own limitations.

4.2.4. Classification of current systems

This section presents currently available legal information extraction systems with a comparison analysis on the scope of the approaches, their limitations in the legal and information science context, and methodologies used to implement the systems. The information gathered here are based on past researches and they are being categorized into several general categories. Each category first explains it's relevant concepts and then does a brief introduction to the system being used.

A. Ontology based legal information extraction

Combining the definitions of information extraction presented by Russell and Norvig [5] and Riloff [81] with a few more factors in consideration, Wimalasooriya et al. [1] present a comprehensive definition for an Ontology Based Information Extraction (OBIE) System as *a system that processes unstructured or semi structured natural language text through a mechanism guided by ontologies to extract certain types of information and presents the output using ontologies.*

Making the general definition abstracting out to the legal domain with slight modifications to suite it to the practical circumstances, an Ontology Based Legal Information Extraction (OBLIE) System could be defined as, *a system that processes unstructured or semi structured natural language text which are related to* *the legal domain through a mechanism guided by legal ontologies to extract certain types of information while preserving heir domain semantic usage and presents the output using ontologies.* However, we keep the fact that presenting the output using an ontology, not as a necessary condition but as a sufficient condition.

Ontology

The definition of ontology is defined as a formal and explicit specification of a shared conceptualization [18, 82]. It should be also noted that most of the ontologies are defined according to a particular domain of interest and since the information extraction is essentially concerned with the task of retrieving information for a particular domain, formally and explicitly specifying the concepts of that domain through an ontology can be helpful [1]. For an example, an ontology on movies could guide the process of information extraction from the user reviews in IMDb [1].

On an eye towards the existing approaches on legal ontologies, Loan Alfred Letia et al. present an ontological approach to the legal literature for translating sources of law into information accessible to people both with and without a legal education [3]. They also highlight the role of two legal ontologies; the LKIF-core Ontology and the Lexis Ontology in improving the dissemination of legal knowledge. Pepijn Visser et al. [2] provide a detailed comparison of four ontologies designed for the use of legal knowledge systems. [13] has presented a novel way for deriving a vector representation for ontology classes with word vector embeddings which in fact can be used in ontology mapping, ontology merging, ontology integration, ontology alignment and semi automated ontology population.

Legal Information Extraction Using Ontology

Ontology based information extraction could be identified as a novel way of extracting information from text corpora. While this approach has shown some promising aspects in many major domains [1], incorporation of ontology based information extraction in the legal domain has less number of efforts compared to other domains. Only a few researches have been carried out on the said motive.

Araujo et al. have exposed a novel way for achieving a more sophisticated system using an ontology, to drive the information extraction process in legal documents [5]. However, albeit the wide use of semantic information described in legal domain ontologies and the integration of linguistic information obtained from the studies of legal documents, the system proposed by Araujo et al. [67] has many practical limitations. The manual term extraction process by using only a selected section of court cases, while limiting that process to only ten court cases, creates a major practical issue in a large legal domain corpora. On the contrary, lack of proper preprocessing in terms of spell checking and grammar checking, could cause a negative effect on the system as well. The heavy coupling and involvement of legal professionals in the linguistic step, which makes the base for the computational step, has also been a major drawback of their proposed system. This high coupling of legal professionals, adds an extra constraint to the scalability of the system. Hence, the system proposed by Araujo et al. [5], while small in scale, works as a proof of concept for the fact that ontology based legal information extraction can make an effective impact on the case of legal information extraction. However, a scalable and extensive efforts are yet to be made.

Araujo et al. [83] have made an another effort to mitigate the limitations of their previous work and coming up with a better ontology based legal information extraction system for juridical events with case studies in the Brazilian legal realm. This work has presented a system for ontology based information extraction from natural language texts, which is able to identify a set of legal events. They have incorporated a domain ontology of legal events and a set of linguistic rules, integrated through inference mechanisms in arriving with the proposed system. This technique has resulted in a flexible and scalable approach for information extraction in the legal domain, while mitigating the practical limitations we saw in their previous work. A satisfactory result in terms of precision and recall have been obtained by the authors through this approach. However, there are future directions for improvements.

B. Rule based information extraction

The rule based information extraction can be identified as a traditional information extraction methodology which is employed in various domains. Nowadays, other information extraction mechanisms which require minimum human involvement and give less recall, have gained much more attention than rule based information extraction. Nevertheless, rule based information extraction is a prominent and much easier method compared to other automated information extraction mechanisms, when extracting information from complicated, structured documents because automated learning of complicated structures, requires enormous

[1]http://www.imdb.com/

amount of data for processing in order to obtain a reliable output. Thus the popularity of this methodology remains as it is in the critical medical domain [41] and as well as in the scientific domain [42]. Several efforts have been taken to combine both rule based information extraction and machine learning based automated information extraction in order to resolve issues in both mechanisms. While traditional approaches for rule based information extraction is primarily based on regular expression grammars, there are attempts on improving the scalability of such methods to large number of rules such as Frederick Reiss at els, algebraic approach to rule based information extraction that addresses these scalability issues through query optimization [43]. However, as described above and according to Laura Chiticarius et al [44], while rule-based information extraction dominates the commercial world, it is widely believed as a dead-end technology by the academia.

The General Architecture for Text Engineering (GATE) is an open source tool which facilitates high level, complex semantic annotations unlike in most traditional rule based approaches. A description regarding this tool is provided in Section 4.2.5.C. GATE facilitates linguists and text engineers to develop and use a wide range of natural language processing tools to analyze a text corpus. Wyner and Peter [39] presents a novel method of rule based information extraction to identify and extract of rules from regulations, in particular, conditional and deontic rules, specifying the antecedents, consequences, agents, themes, actions and exceptions by extending the above approach.

Wyner and Peter [39] have employed standard natural language processing components such as tokenizing, sentence splitting, part of speech tagging, lemmatizing, parsing etc., which are fed into GATE in a form of a pipeline. Java Annotation Patters Engine(JAPE) rules are used to create complex rules using annotation and regular expressions and then used to subsequently output an annotated text corpus.

Comparing the results of the final system against a golden standard (described in Section 4.2.6.A, while the system has output a promising result for most situations, some of the situations have performed poorly, proving poor recall bottleneck, which emerges in most of the rule based information extraction systems. Every result shows favorable precision above 85% and indicates that rule based information extraction systems are capable of handling precision critical information extraction systems.

In the system developed by Walter and Pinkal [45], it describes a rule based information extraction technique which extracts definitions from legal court decisions. This system is being evaluated on a corpus of about 6000 German court decisions within the field of environmental law. By conducting a set of surveys on randomly selected verdicts, they have managed to identify some common structural elements found in legal definitions. This approach tries to use natural language processing techniques to create rules in order to extract relevant information from a given piece of text. Similar researches being carried out for this purpose, are the systems proposed by Lame in 2005 [46], and Saias et al. in 2005 [47], where both of these fall under information retrieval.

In the system proposed by Walter and Pinkal [45], the general structure of the definitions of legal definitions, are being modeled using the survey results. The limitation in this is that the survey has only covered a limited number of definitions whereas the same definition can be represented in different structures in different on-line resources. Therefore the scope becomes limited to the definitions in interest and of the selected data source. This research has developed 33 such extraction rules, which gives a 50% hit rate and a precision of well above 70%. The selection of rules and filters have been done based on the optimization of precision for the evaluation of the system. An example extraction rule is given in Figure 4.3.

```
<pattern>
description=liegt vor + wenn-Nebensatz
query=sent/parse/preds/word[@stem="vorliegen"
        and INDPRES and WENN]
filters=definite
definiendum=DSub
definiens=WENN/arg/word
area=PPMOD{PREP%bei}
</pattern>
```

Figure 4.3: Extraction Rule Example

Even though precision gives an acceptable result, recall has not been addressed in this system [45]. This is mainly due to the difficulty in finding proper legal corpora with annotated definitions in the legal domain, and building of such corpora is very time intensive. It is also clear that a considerable amount of definitions cannot be identified by purely linguistic features.

C. Machine learning based legal information extraction

As explained in the introduction, many researchers have suggested different approaches to automate information extraction from legal documents such as acts, constitutions and statutes etc. Approaches which use human involvement are difficult to scale into a large systems since they are time consuming and tedious tasks resulting low recall in the final output. A fine explanation on how knowledge discovery and information extraction is being carried out using data mining and machine learning techniques, is mentioned in "Knowledge Discovery from Legal Databases" [84] written by Stranieri and Zeleznikow.

The learning problem can be described as coming up with a general function that will explain the data, given a sample of limited size. This sample is known as the set of training samples, that will be used to train the machine learning model being built. The research carried out by Paulo and Teressa on identifying entities from judicial documents using linguistic information and machine learning techniques [48] which uses support vector machines for this purpose.

Text Classification

In the case of machine learning, if the output space has no structure except whether two elements are equal or not, we have a classification task. The purpose of text classification is to classify each input object into a predefined class in the system. Each element of the output space is called a class. The supervised classification task of natural language texts is known as text classification [48].

This text classification task has being a very dominant research area in the machine learning field. Algorithms such as decision trees [49], linear discriminant analysis [50], naive Bayesian algorithms [51] and support vector machines [52].

Support Vector Machines

Support Vector machines is a method introduced by Cortess and Vapnick in 1995 [85] and this uses a Kernel technique that consists of two main parts.

- Kernel function
- Learning algorithm

The kernel function does an implicit mapping of the non-linear pattern of the data into a vector space called the linear feature space. This depends in the data type being used and the domain knowledge of the particular data source.

The learning algorithm, is implemented in a way that the coordinates of the embedded points in the linear feature space are not needed; only their pairwise inner products. According to Shawe-Taylor and Christianini [86], this algorithm is said to be robust and efficient since the amount of computational resources required remains polynomial, even when the size of the linear feature space grows exponentially. The support vector machine classification general idea is depicted in Figure 4.4.

The reason for Paulo and Teressa [48] to go forth with the use of Support Vector Machines (SVMs) for text classification is that in [87], it says that using SVMs to learn text classifiers is the first approach that is computationally efficient, and performs well and robustly in practice. There is also a justified learning theory that describes its mechanics with respect to text classification. The methodology, uses a support vector machine classifier to associate concepts to legal documents and a natural language parser to identify named entities, namely, locations, organizations, dates, and references to other articles and documents.

According to C.Biagio et al [88], legal text is a composition of formal partitions (articles, paragraphs etc.) or by semantic units containing fragments of a statement in a law that particular thing must happen or be done (provisions). Semantic annotation of a legal text makes retrieval of norms easier. An automated mechanism of detecting provision types and descriptions using multi class support vector machine algorithm has been described by C.Biagio et al in [88]. Support vector machine methodology for multi class classification tasks is implemented either by combination of binary support vector machine classifiers or by direct implementation of multi class SVM learning algorithm. Authors have selected the second approach of implementing multi class support vector machine which has resulted a promising precision and recall in the system.

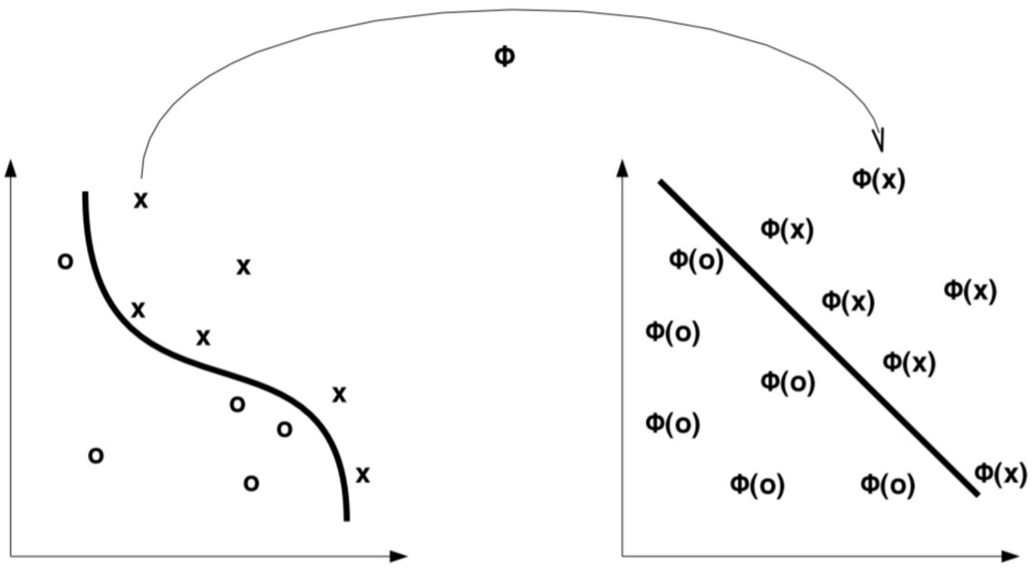


Figure 4.4: Support Vector Machine Classification

Identification of Legal Concepts using SVM

For this purpose, Paulo and Teressa [48] has used the “bag of words” technique [53] in order to carry out the preprocessing of the textual representations, and it is being run on SVM light [54] with a linear kernel and other default parameters.

As the first step, this approach uses the Bag of Words technique to represent each document being used. It maps all the numbers to the same token, and use the tf-idf weighting function [59] given in Equation 4.1, to normalize the unit length.

$$tf-idf(w_i,d) = tf(w_i,d)ln\frac{N}{df(w_i)} \tag{4.1}$$

Afterwards, it uses the Stratified Cross Validation technique [55], which is a model evaluation method that divides the original dataset into subsets, each with the same distribution of examples, between categories as the original dataset [48]. Then one of the subsets is used and the remaining subsets are used to form a training set. This procedure is repeated for each subset separately and the variance of the resulting estimate is reduced as the number of subsets increase.

But, the system proposed by Paulo and Teressa [48] has the same problem that we saw in the Section 4.2.4.B under the Rule Based technique. In the legal concept identification, this system shows good results in terms of precision but does not show much good results in terms of recall. In terms of named entity recognition, the referencing does not have a number of unique references and it requires further text processing of the documents. According to the evaluation, the concept identification task has a better precision for English, German, Italian and Portuguese, whereas worst results were obtained for Romanic languages. The named entity task has shown good results for date identification, but bad results for identification of organizations and references to other articles and legislation.

Neural Networks

Neural networks in computer science (Artificial Neural Networks) are inspired by biological neural networks that comprise in brains of animals. Artificial Neural Networks learn to do tasks by examining and learning training examples. For instance, in text categorization, news articles which are manually categorized into “sports” and “not sports” have to be fed into neural network in order to learn, and then the trained neural network may identify sports news from given set of news articles. These neural networks are being employed in complex information extraction systems which can’t be addressed using traditional rule based information extraction systems.

Artificial Neural Networks are made up by connected base units called perceptrons, which are organized into layers. Signals travel from input layer to output layer through passing layers(hidden layers) in between

,and weights accompanied with connections between perceptrons together convert the input into output accordingly.

Laurent, et al [89] has conducted an experiment towards extraction of legal decision rules by occupying a multilayered neural network. Thus, extracted rules are used for designing the knowledge base of an expert system. The neural network has been used as a classifier whose target is a legal rule. The network can be made to produce its knowledge in the form of explicit rules.

Extracting information from criminal-justice data which are written in natural language text such as police narrative reports is a laborious tasks and vital task for investigators in crime analysis. Michael C, Junnifer J. et al [90] have developed a name-entity extraction technique based on neural network to identify useful entities from police narrative reports. Proposed system has performed well for recognizing person names and narcotic drugs. Evaluations done in small-scale shows that the systems will work real world applications.

Semi-Supervised Information Extraction Systems

Semi-supervised learning falls between unsupervised learning and supervised learning that makes use of fewer labeled data than trivial supervised machine learning algorithms. Labeling data is often an infeasible laborious task and requires skilled human agents, which leads to high cost involvement. Semi-supervised learning is a notable alternative in order to minimize labeling cost with a considerable retention of accuracy.

A novel method for extraction of textual features has been developed by Tianhalo W. and William M. [91] with the aim of aiding legal authorities to investigate police incident reports, university crime reports and patents. Proposed system has performed competitively when compared to other extraction systems which were used in criminal justice information extraction systems. The proposed system employs a semi-supervised active learning algorithm for pattern recognition that consists of regular expressions. The approach is semi-supervised since it does not require the precise locations of features in the training data which ultimately saves lots of human effort. Algorithms are seeded with a single positive example of a given attribute to discover the candidate attributes which are then fed into the manual pruning mechanism.

Ontology-based information extraction has emerged in the last decade as a subsection of information extraction and ontologies, are increasingly utilized in many heterogeneous fields for knowledge representation. Here, Information extraction is usually guided by ontologies and the final output is presented through an ontology. Vindula J. and Lakmal D. et al [14], emphasize that traditional methodologies of ontology population has a problematic bottleneck of heavy involvement of human intervention. They have implemented a semi-supervised instance population of a legal ontology using word vector embedding. They have implemented several traditional benchmark models of ontology population, along with novel methods, based on word embeddings and ensemble all the models based on accuracies of each model that was measured according to a golden standard, to come up with a synergistic model with better accuracy.

D. Template based information extraction

Template based information extraction systems, extracts information from a given piece of text and creates them as a template for further processing. These templates can be later used to match with existing databases to check whether the extracted information is matching to the information available on the database. There are many systems implemented to extract information from text and many systems to build links to relational database records.

Once the information is extracted from the text, the system will automatically create a template using the data available. Some systems use techniques that do not use templates, to search the databases. Some systems do a one to one mapping between the keywords found and the relational database records where a possible keyword search is being carried out directly, with the tags available in the database records. There are other systems which use statistical approaches to match the data in the text and information available in the databases. These statistical methods involve probabilistic approaches where the probability is measured for the equivalence of the text being extracted and the information in the database.

The system taken as an example in this classification of information extraction, is the system developed by the Computer Science Research Department of West Group. The goal of this implementation is to extract person names from legal text and link them with a pre-saved biography of each of the persons automatically.

Automatic extraction and linking of person names in legal text

This is an application that creates hypertext links in texts, from named individuals to personal biographies [73]. The system creates MUC-style templates [92] from legal texts and linking them to biographical data of persons in a relational database. The linking technique is based on a naive Bayesian inference network. Although there are systems that extract data from text and systems that link to relational database records, it is highlighted that there is no such system to automatically generate hypertext links. In the experiments being carried out by this system, it was able to extract and link attorney names in attorney paragraphs with 0.99 precision and 0.92 recall when compared to links created manually. It was also able to extract and link judge names with 0.98 precision and 0.90 recall.

The extraction portion of this system is similar to other template extraction systems described in Message Understanding Conferences proceedings (MUC-6, 1995) [64]. This extraction process relies on a finite state machine [93] that identifies paragraphs in case law containing attorney and judge names and a semantic parser that extracts attorney and judge template information from the paragraphs.

The overall system consists of four main components.

1. Extracts attorney and judge templates from case law.
2. Matches the extracted templates to biography records which have id numbers.
3. Inserts hypertext links into case-law using the id numbers of the people identified by the matching module.
4. Loads the biographical records in the relational database to the West's IR system as text.

Template generation

The template generation happens for each identified name in the paragraphs. Once this template is generated, it uses a naive Bayesian network to identify the probabilities for the completion of matching of templates with person biographies. An example template is given in Table 4.1.

Table 4.1: Generated Template

Last Name	Parrish
First Name	Robert
Middle Name	B.
Name Suffix	Blank
Firm	Moseley, Warren, Prichard & Parrish
City	Jacksonville
State	FL
Date	June 22, 1998
Paragraph offset	75
Name Length	17

E. XML Based Information Extraction

XML(Extensible Markup Language) is a markup language to define a set of encoding rules which are both human and machine readable [94]. Even though XML was originally designed to represent formal documents, it is now widely used to represent various data structures and message passing in web services. XML has become popular in no time as it is able to use as a platform independent exchange mechanism and it is able to handle sharing data between programs without prior coordination.

XML is considered as a key adaptation for modeling and representing legal documents in recent past. XML provides the feasibility to model any structure which is very much appreciated in dealing with legal documents [95]. According to Fabio Vitali et al [96], the possibility to obtain hypertext functionalities with XML is another key factor why XML is appraised as a feasible language for modeling legal documents. Furthermore Andrea Marchetti et al [97] emphasize the fact that ability to manipulate the semantic features in XML format is a prominent quality in the case of XML format utilization in legal domain.

Gloria T. Lau, Shawn Kerrigan et al [98] have developed a government regulation analysis and compliance assistance system exploiting benefits of XML as a data structure such as capability of hierarchical representation and handling semi-structured data. The complexity and broad amount of sources of heterogeneous government regulations has made understanding and representing government regulations difficult. Authors have proposed an information infrastructure which comprises a document repository built, based on XML format to facilitate regulation analysis.

M.Mercedes Martinez et al [99] has approached the use of XML as for means of legal information extraction as a part of research project aimed at manipulating Spanish legal documents. Their experiment were made on Spanish normative documents in terms of laws, decrees, royal-decrees and circulars. The experiments were based on two major steps; preprocessing alongside manual revision to generate XML documents, and the structure extraction process on the document set obtained after the preprocessing step.

F. A Mixed Approach

Most of the methodologies outlined above have competencies as well as drawbacks resulting drops of accuracy and reliability of extracted information. Several attempts have been taken to combine above methodologies aiming elevating the quality of information extraction.

Paulo Quaresma has addressed this approach through combining machine learning methodologies with linguistic information techniques. Lexical, syntactical and semantical information extracted using linguistic information techniques are fed as inputs into specialized machine learning algorithms such as support vector machines. Natural language processing tools are used to analyze legal texts and to obtain lexicons with parts of speech tags, syntactical parse trees, and partial semantical representation linked with an ontology. This linguistic information is fed into machine learning algorithms which is responsible for high-level information tagging and extraction. The machine learning algorithms will improve the results of the named entity recognition (NER) task, which is basically the process of identification of persons, organizations, places etc. Moreover, it proposes to use the entities identified in the extraction process to automatically populate an ontology for easier implementation of knowledge based systems. Furthermore, authors have claimed that the results obtained in preliminary experiments were quite promising for legal text collections.

4.2.5. Implementation

In general, information extraction systems follow three different steps in operation [66].

1. Textual Analysis
2. Extraction Rules Selection
3. Extraction Rules application and Results Identification

Textual analysis contains the analysis of given text documents where shallow natural language processing techniques [100] can be used, to segment, tokenize, lemmatize and so on. The second step is to identify the relevant format and structure of the legal document, in order to identify the conditions and rules that can be generated for the information extraction process. The final step is more into evaluations and validate the results. Here, the application of the extraction rules will also be considered for current and future implementations.

When we look at current approaches used for domain specific information extraction, the text analysis aspect plays a major role. Many enhancements can be made with simple preprocessing techniques which can be incorporated with linguistic measures as study shown by Sugathadasa et al [101]. This, in-line with domain specific knowledge and knowledge models, the task of informaiton retrieval can be made more accurate than existing methods. In general, these have become commonly adopted techniques by many researchers who address the case of domain specific information extraction. The following sections elaborates the tools and techniques used in the implementation phase of each legal information extraction system.

A. Text Analyzing Tools

Linguistic techniques in the field of information extraction has become a common approach as it can better enhance the final results of the extraction process. This solely depends on the text corpora being used, and the domain in analysis. Some domains do not really need linguistic preprocessing whereas some domains heavily rely on preprocessing to make the results successful.

One of the major tools are the shallow natural language processing tools being used. These tools can perform functions such as Parts of Speech (POS) tagging, sentence splitting, lemmatizing, identifying occurrences of regular expressions and tokenizing [1]. Traditionally, deep linguistic processing [102] involves linguistic grammar development and execution for various tasks. But, in the recent years, machine learning techniques have boosted up natural language processing (NLP) techniques, which requires very less manual labour unlike in deep linguistic processing.

One of the major shallow NLP tools is the Stanford Corenlp natural language processing toolkit [103]. This tool provides a wide range of shallow NLP techniques such as tokenizing, sentence splitting, parts-of-speech tagging, morphological analysis (lemmatizing), named entity recognition, syntactic parsing, co-reference resolution, and sentiment analysis.

Another major tool for text analysis is the General Architecture for Text Engineering (GATE) [104], which is a Java Suite, where worldwide scientists, students, companies, and teachers use to execute their natural language processing tasks as required. This tool includes an information extraction system named ANNIE (A Nearly New Information Extraction System), that provides a variety of tools such as a tokenizer, a gazetteer, a sentence splitter, a part of speech tagger, a named entities transducer and a coreference tagger. One major advantage why developers and researchers prefer this is that it supports multiple languages [105].

Some of the other common NLP tools are presented in a survey study [106], carried out by Reeve and Han. AeroDAML [107] is designed to map proper nouns and common relationships to corresponding classes and properties in DAML ontologies [108]. Armadillo [109] uses a pattern based approach to extract entities. KIM (Knowledge Information Management) [109] platform contains an ontology, a knowledge base, a semantic annotation, an indexing and retrieval server.MnM [110] uses an ontology based information extraction tool that uses Lazy NLP algorithms. MUSE [111] uses the GATE framework to perform named entity referencing and coreferencing. Ont-O-Mat [112] is a semantic annotation framework which is machine learning based. SemTag [113] is the semantic annotation component of a comprehensive platform, called Seeker, for performing large-scale annotation of web pages.

B. Semantic Lexicons

Most text processing systems utilize distributed and statistics based models in order to perform various natural language processing (NLP) techniques. These models are backed up by semantic reference documents or dictionaries that can guide the nlp technique. These dictionaries are known as semantic lexicons which is a digital dictionary of words which are labeled with semantic classes so that associations can be drawn between words that have not been previously encountered [114]. These are also known as lexical-semantic databases and lexical-semantic nets [1], which helps words to organize themselves accordingly according to the semantics, syntax and the relationships between words.There are different kinds of lexicons that can cater for specific languages and domains. Some popular examples are WordNet [115], EuroWordNet [116], GermaNet [117], Multilingual Central Repository [118], Global Wordnet [119], MindNet [120] and Hindi WordNet [121]. There are over thousands of different lexicons that can be selected from, for different purposes and domains.

C. Linguistic Rule Extractors

A linguistic rule is a rule which is written to separate different portions of a given piece of text. In most cases, this can be achieved using regular expressions which will identify and extract certain portions of the text. Regular expressions are a powerful, flexible and efficient text processing pattern that uses a general pattern notation to describe and parse text [122]. Regular expressions use shallow Natural Language Processing (NLP) tools like Parts-of-Speech tagging, tokenizing etc. With these tools, the linguistic rules provide the ability to extract a significant amount of information where regular expressions is often implemented using finite state transducers which consists of a finite state automata [1].

The General Architecture for Text Engineering (GATE) [104], is a powerful NLP tool, that provides a common infrastructure for building language engineering systems. Information extraction is one of the major tasks that it can perform, where GATE provides a platform to evaluate, reconfigure, and reevaluate the modules generated through GATE. Providing rule based information extraction is one of the major features where it introduces a platform to extract certain parts of the text and validate them, with the use of rule generation. These rules are referred to as JAPE (Java Annotation Pattern Engine), where it provides finite state transduction over annotations based on regular expressions [123]. Many information extraction systems have been developed with the help of JAPE rules. Maynard et al. [124] proposed a rule based system to extract market monitoring system. Wyner and Peter [125] incorporates JAPE rules in extracting legal case factors from legal cases. Wyner also carried out a similar research to annotate and extract legal case elements using JAPE rules [126]. A system to carry out legal text summarization by exploration of the thematic structures and argumentative roles, also uses JAPE rules to develop semantic grammar rules [127].

FASTUS Information Extraction System [93], is a finites state processor for information extraction from real world texts, that can be used for this purpose. This tool has not been necessarily used for the legal domain, but the e-government domain has used this for semantic information extraction [128] and other similar domains.

D. Web Crawling and Text Corpora

Finding relevant and appropriate text corpora off the digital repositories available on-line is very complicated and time consuming. This is mainly due to the lack of consistency and availability. Web crawling is an approach that allows to collect relevant legal documents from the web. JSoup [129], Beautiful Soup [130], and Selenium Web Driver [131] are some of the widely used web crawlers.

Text corpora is used as an evaluation technique for information extraction systems. Especially for the legal domain, the text corpora is being used to validate the information extraction systems being developed, to see whether they extract the relevant information accurately. GATE [104] provides a tool for rule based information extraction and validation mechanism using a given text corpus.

4.2.6. Performance Measurements

In the field of information extraction (IE), Precision and Recall are two widely used performance metrics, that can eva;uate the performance of these systems. As per Wimalasooriya at el., precision shows the number of correctly identified items as a proportion of the total number of items identified, while recall shows the number of correctly identified items as a proportion of the total number of correct items available [1]. Adhering to the same definition, precision and recall in legal information extraction systems could be identified as well. Following formulas illustrate the definition pertaining to legal information extraction where $Relevant$ denotes the set of relevant documents and $retrieved$ denotes the set of retrieved documents.

$$Precision = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Retrieved\}|}$$

$$Recall = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Relevant\}|}$$

On a different note, the same definitions can be illustrated with a more abstract way by incorporating a golden standard. Semantic similarity measures built for general purpose use, do not perform well within specific domains. Hence, the domain specific semantic similarity measure that was created by the synergistic union of word2ve; a word embedding method that is used for semantic similarity calculation and lexicon based (lexical) semantic similarity methods have been introduced by Sugathadasa at el. [101].

$$Precision = \frac{|\{Correct\ Retrieved\ Documents\}}{|\{All\ Retrieved\ Documents\}|}$$

$$Recall = \frac{|\{Correct\ Retrieved\ Documents\}\||}{|\{All\ Documents\ in\ Golden\ Standard\}|}$$

However, precision and recall often has a trade-off between each other. That is, search strategies can be adjusted to increase precision at the expense of recall, or vice versa. For an example, 100% recall could be achieved by retrieving entire document population at the cost of precision. On the other hand, 100% precision could be achieved by retrieving a single relevant document at the cost of recall. A Precision-Recall Curve illustrates the Precision-Recall Trade-off for a particular search method [132]. In mitigating this effect, F-measure is often used together with precision and recall.

$$F - Measure = \frac{(\beta^2 + 1)\ * Precision * Recall}{(\beta^2 * Precision) * Recall}$$

Here, β denotes the weighting of precision vs recall. In most situations, 1 is used for β, giving equal weights for precision and recall. This is known as F1 score.

Some other metrics such as Slot Error Rate, SER [133] is also used as a performance evaluation metric in typical IE systems.

$$SER = \frac{incorrect + missing}{key}$$

Here, *key* denote the total number of slots expected to be filled according an annotated reference corpus, *incorrect* be the number of incorrectly filled slots in the system's response and *missing* denotes the number of slots in the reference that do not align with any slots in the system response.

A. Golden Standard

The golden standard is a model which is accepted to be accurate, in a selected domain of interest. The golden standard can be modeled in different ways, which are briefly explained below. In general, the golden standard is developed with the help of domain experts which can be further used to assist the performance measuring aspects of text mining tasks. a novel approach in generating a golden standard without the help of legal domain experts, is given below as well.

A golden standard for legal information extraction systems can be defined in different ways. Sugathadasa et al. [101] has addressed this problem with the help of domain experts, by creating a vocabulary of similar words to a selected set of words used in their study. This approach is time consuming, where more human effort is needed for better accuracies.

Similarly, the use of ontologies as a guidance for information extraction is an upcoming research area, where Araujo et al. [5] uses legal domain experts coupled with the information extraction process and the generation of the ontology. Here, the legal documents were annotated and used in the research, which was based as the golden standard in their process. Their heavy coupling with legal domain experts made the entire process limited in scope, where further enhancements needed heavy human labor requirements.

In a recent study, a novel approach was proposed for generating a golden standard, with the help of text mining techniques. The coupling of domain experts in this study is minimum, where the concept of mention map was introduced.

4.3. Ontologies

Ontologies are mainly used to organize information as a form of knowledge representation in many areas. As defined by Thomas R. Gruber [18], 'ontologies are an explicit and formal specifications of the terms in the domain and the relations among them'. Ontologies have been expanding out from the realm of Artificial-Intelligence to domain specific tasks such as: Linguistics [10, 12, 20, 22, 134], Law [13], Medicine [8, 23, 24]. Ontologies have become common on the semantic iteration of the World-Wide Web. An ontology may model either the world or a part of it as seen by the said area's viewpoint [20]

The basic ground units of an ontology are the *Individuals* (instances). By grouping these *Individuals* which can either be concrete objects or abstract objects, the structures called *classes* are built. A *class* in an ontology is a representation of a concept, type, category, or a kind. However, these definitions may be altered depending on the domain of the ontology. Often these *classes* form taxonomic hierarchies among them by subsuming, or being subsumed by, another class.

4.4. Lexical Semantic Similarity Measures

Applications of text similarity measures include, relevance feedback classification [135], automatic text summarization by text restructuring [136], automatic evaluation of machine translation [137], semi-supervised ontology population [14], and determining text coherence [138]. Some approaches are based on statistical methods [139], vector representations, string/corpus based approaches, and hybrid similarity measures where four similarity measures were tested in [101] and eight similarity measures were tested in [140].

TF-IDF alone uses an inverse document frequency for term frequencies where it does not consider surrounding context amongst words in a text. This mapping is simply done using count and probabilistic measures.

Lexical Semantic similarity of two entities is a measure of the likeness of the semantic content of those entities, most commonly calculated with the help of topological similarity existing within an ontology such as WordNet [21]. Wu and Palmer proposed a method to give the similarity between two words in the 0 to 1 range [15]. In comparison, Jiang and Conrath proposed a method to measure the lexical semantic similarity between word pairs using corpus statistics and lexical taxonomy [16]. Hirst & St-Onge's system [17] quantifies the amount that the relevant synsets are connected by a path that is not too long and that does not change direction often. The strengths of each of these algorithms were evaluated in [12] by means of the tool WS4J[2].

[2]https://code.google.com/p/ws4j/

4.5. Vector Similarity Measures

In the field of information retrieval, the similarity between documents or terms are measured by mapping them into a vector of word frequencies in a vector space and computing the angle between the pair of vectors [141]. Distance and similarity measures encounter in various fields like chemistry, ecology, biological taxonomy and so on. Some of the traditional approaches used for similarity/dissimilarity measures include Hamming Distance [142, 143], inner product, Tanimoto distance etc. Sung-Hyuk Cha [144] addresses similarity measures available in terms of both semantic and syntactic relationships which are being used in various information retrieval problems. Sung-Hyuk Cha et al. [145] also describes how vector similarities can be obtained by enhancing binary features on vectors. Jayawardana et al. has utilized vector similarity measures in deriving representative vectors for ontology classes [13].

The word similarities are calculated according to the distance in space between two vectors. Wu and Palmer proposed a method to give the similarity between two words in the 0 to 1 range [15], where they have further used the cosine distance [146] as a measure of similarity.

The cosine similarity is measured by the angle between the two vectors. If they are parallel, the cosine distance is equal to one and the vectors are said to be equal. If the vectors are perpendicular, they are said to be dissimilar (no relation to each other). This similarity is based on terms or words in the legal domain. For example, it will show that words like father, mother, family are falling as similar words whereas tree, book and king falling as dissimilar words.

Similarly, the relationship between vectors can be measured in terms of the displacement between the two vector points. This is really helpful when the relationship between two elements is not known. For example, let's assume that we know the displacement between the vectors of man and woman, and the vector direction. Then if we need to find a similar relation for king, the model will find a vector in the same direction with a similar displacement from the point of the king vector, and will return queen as the answer.

The similarity and relationships between legal terms are very complex, where some relationships are based on hierarchical models. By the vector space obtained via word2vec, our study can prove many similarities and relationships which are not possible in a general word2vec implementation [9]. With this, we intend to show that results tend to be more accurate when the entire process is integrated with NLP techniques as described in this study.

4.6. Legal Information Systems

One of the leading domains, that severely adhere to this issue is the medical profession. The Canon group has come up with a representation language [147] for medical concepts in 1994, whereas word representations for hospital discharge summaries [148] was developed in 2013.

Schweighofer [149] claims that there is a huge vacuum that should be addressed in eradicating the information crisis that the applications in the field of law suffer from. This vacuum is evident by the fact that, despite being important, there is a scarcity of legal information systems. Even though the two main commercial systems; WestLaw[3] and LexisNexis[4] are widely used, they only provide query based searching, where legal officers need to remember keywords which are predefined when querying for relevant legal information. Hence, there is still a hassle in accessing this information.

One of the most popular legal information retrieval systems is KONTERM [149], which was developed to represent document structures and contents. However, it too suffered from scalability issues. The currently existing implementation that is closest to our proposed model is Gov2Vec [150], which is a system that creates vector representations of words in the legal domain, by creating a vocabulary from across all corpora on supreme court opinions, presidential actions and official summaries of congressional bills. It uses a neural network [151] to predict the target word with the mean of its context words' vectors. However, the text copora used here, itself was not sufficient enough to represent the entire legal domain. In addition to that, the Gov2Vec trained model is not available to be used by legal professionals or to be tested against.

There are certain semantic and syntactic relationships, which are very specific for a domain like law, as it is to other domains like medicine and astronomy. Even though they point out that these relationships can be included in the training process, it seems like there is a lot more work to be done due to the special syntactic and semantic behavior in the legal domain.

Languages being used are sometimes mixed up with several origins (i.e English, Latin etc) and in certain

[3]https://www.westlaw.com/
[4]https://www.lexisnexis.com/

cases, the meaning of the words and context differs by the legal officers' interpretations. One of the popular systems named KONTERM [149], is an intelligent information retrieval system that was developed to represent document structures and contents, to address this issue.

This is where Bag Of Words techniques, along with sentence similarities, play a major role in not only coming up with a representation scheme for words and phrases using a vector space, but also in identifying semantic similarities between the documents, which can be used in many applications for context identification and other purposes.

4.7. TextRank Algorithm

TextRank algorithm [152], is based on Google's *PageRank* algorithm [153]. *PageRank* (PR) was used by Google Search to rank websites in their search engine results. *TextRank* uses *PageRank* to score the sentences in a document with respect to other sentences in the document by using the graph representation G, for the set of sentences S, where the edges in set E represent the similarity between each of these sentences in the given document.

$$G = \{S, E\} \tag{4.2}$$

The first step is to separate the document into sentences to be used by TextRank. Next, a bag of words for each of the sentences is created. This is an unordered collection of word counts relevant to each sentence. This creates a sparse matrix of words with counts for each of them. Then a normalization technique is used for each count, based on TF-IDF. The next step is to create a similarity matrix A between sentences, which shows the relevance and similarity from one sentence to another. It is then used to create a graph as shown in Figure4.5. A threshold is used to decide which edges would stay in the final graph.

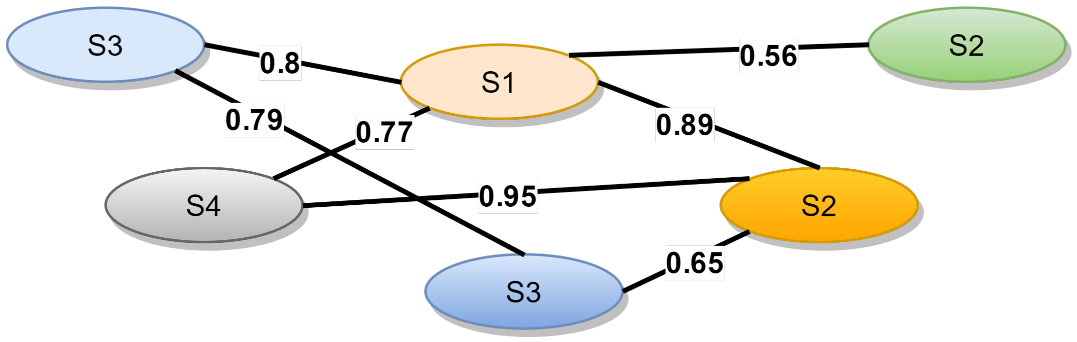


Figure 4.5: Sentence Similarity Graph Network

Finally, the PageRank algorithm will be used here to score the edges of the graph.

4.8. Word Vector Embedding

Traditionally, in Natural Language Processing systems, words are treated as atomic units ignoring the correlation between the words as they are represented just by indices in a vocabulary [9]. To solve the inadequacies of that approach, distributed representation of words and phrases through word embeddings was proposed [25]. The idea is to create vector representations for each of the words in a text document along with word meanings and relationships between the words all mapped to a common vector space.

A number of Word Vector Embedding systems have been proposed such as: GloVe [154], Latent Dirichlet Allocation (LDA) [155], and word2vec[5] [9]. GloVe uses a *word to neighboring word* mapping when learning dense embeddings, which uses a matrix factorization mechanism. LDA uses a similar approach via matrices, but the concept is based on mapping words with relevant sets of documents. word2vec uses a neural network based approach that uses *word to neighboring word* mapping. Due to the flexibility and features it provided in terms of parameter passing when training the model using a text corpus, we use word2vec in this study. word2vec supports two main training models: Skip-gram [156] and Continuous Bag Of Words (CBOW) [9].

[5]https://code.google.com/p/word2vec/

4.9. Word Set Expansion

Word lists that contain closely related sets of words is a critical requirement in machine understanding and processing of natural languages. Creating and maintaining such closely related word lists is a complex process that requires human input and is carried out manually in the absence of tools [20]. The said word-lists usually contain words that are deemed to be homogeneous in the level of abstraction involved in the application. Thus, two words W_1 and W_2 might belong to a single word-list in one application, but belong to different word-lists in another application. This fuzzy definition and usage is what makes creation and maintenance of these word-lists a complex task.

De Silva et al. [20] describe a supervised learning mechanism which employs a word ontology to expand word lists containing closely related sets of words. This study has been an extension of their previous work [10], which was done to enhance the refactoring process of the RelEx2Frame component of OpenCog AGI Framework, by expanding concept variables used in RelEx.

4.10. Ontology Population

Being a knowledge acquisition task, ontology population is inherently a complex activity. Ontology population has been approached by using techniques such as rule based and machine learning. SPRAT [157] combines aspects from traditional named entity recognition, ontology-based information extraction, and relation extraction, in order to identify patterns for the extraction of a variety of entity types and relations between them, and to re-engineer them into concepts and instances in an ontology. Rene Witte et al. [30] has developed a GATE resource called the OwlExporter, that allows to easily map existing NLP analysis pipelines to OWL ontologies, thereby allowing language engineers to create ontology population systems without requiring extensive knowledge of ontology APIs.

However, modern day recherches are more focused on semi supervised ontology population due to the nature of less manual intervention.

4.11. Semi Supervised Ontology Population

Although supervised machine learning methodologies have showed promising results when it comes to information extraction, they accumulate more cost for training since they require vast number of labeled training data. As a solution, semi-supervised machine learning methodologies have been introduced, requiring considerably less amount of labeled training data.

Carlson [158] proposed a semi-supervised learning model to populate instances of a set of target categories and relations of an ontology by providing seed labeled data and a set of constraints which couples classes and relationships of an ontology. Semi-supervised algorithms tend to show unacceptable results due to 'semantic drift' and constraints have been introduced to overcome the issue. Carlson has used 'Bootstrapping' method for semi-supervised learning which starts with a small number of labeled data and grows labeled data iteratively, which are chosen from a set of candidates, which is classified using the current semi-supervised model. Three types of constraints have been introduced by Carlson to conform mutual exclusion, type checking, and text features.

Carlson [159] has expanded coupled semi-supervised learning [158] to never-ending language learning (NELL); an agent that runs forever to extract information from the web and populate them continuously into a knowledge base. A prototype of the system that they have implemented is able to extract noun phrases related to various semantic categories, and semantic relations between categories. Its information extracting ability increases day by day which is evidenced by the ability to extract more information from previous day's text sources more accurately. Input ontology in the system was included with seed instances for each ontology class and then sub systems which consist of previously described coupled semi supervised methodologies extract candidate instances and relationships from the text corpus. Knowledge Integrator of the system choose strongly supported sets of instances and relations from the candidate set, as new beliefs of the system.

Zhilin Yang [160] has presented a semi supervised learning methodology based on graph embeddings. The system consists of two main sections namely 'transductive' and 'inductive'. The 'transductive' approach predicts instances which are already observed in the graph in the training period. In 'inductive' approach, predictions can be made on unobserved instances in the training period. A probabilistic model was developed to learn node embeddings to generate edges in a graph.

4.12. Clustering

Clustering is a seminal part in exploratory data mining and statistical data analysis. The objective of clustering is to group a set of items into separate sub-sets (clusters) where the items in a given cluster is more similar to each other than any of them is similar to an item from a different cluster. The used similarity measure and the desired number of clusters are application dependent. A clustering algorithm is inherently modeled as an iterative multi-objective optimization problem that involves trial and error which tries to move towards a state that exhibits the desired properties. Out of all the clustering methods available, we selected k-means clustering due to the easiness of implementation and configuration.

Arguably, k-means clustering was first proposed by Stuart Lloyd [161] as a method of vector quantization for pulse-code modulation in the domain of signal processing. The objective is to partition n observations into k clusters where each observation i belongs to the cluster with the mean m such that in the set of cluster means M, m is the closest to i when measured by a given vector distance measure. It is implicitly assumed that the cluster mean serves as the prototype of the cluster. This results in the vector space being partitioned into Voronoi cells.

4.13. Support Vector Machines

Support Vector Machines [85] is a supervised learning model that is commonly used in machine learning tasks that analyze data for the propose of classification or regression analysis. The SVM algorithm works on a set of training examples where each example is tagged as to be belonging to one of two classes. The objective is to find a hyperplane dividing these two classes such that, the examples of the two classes are divided by a clear gap which is as wide as mathematically possible. Thus the process is a non-probabilistic binary linear classifier task. The aforementioned gap is margined by the instances that are named *support vectors*.

In Generic SVM, new examples that are introduced are then predicted to be falling into either class depending on the relationship between that new example and the hyperplane that is dividing the two classes. However, in this study we do not need to employ the new instance assignment. We are only interested in calculating the support vectors.

5 Technical Literature

This section gives a brief introduction to the technical background of our research and the areas that we have covered so far in our study. This technical literature has been used in many parts of our research as support and guidance in understanding the concepts better.

5.1. Web Crawlers

A web crawler is a software bot (Internet bot) that will surf through the world wide web in a systematic manner and collects required information in a meaningful way. Typically it's purpose is to do web indexing (web spidering). Some web crawlers collect indexes of various resources and the collected information can be saved in a usable way. There are also some systems which use web crawlers to create a copy of all the visited pages, for later processing by a search engine, that will index the downloaded pages to provide faster searches.

Crawlers often consume a lot of resources on the systems they visit and most crawlers do visit these sites without a consent of the web page. Issues regarding to scheduling, load balancing and 'politeness' comes into play whenever web crawlers visit large collections of pages available online. Sometimes there are public websites that do not like web crawlers since they will consume a lot of resources and power of the servers being used by each of the web pages. For these reasons, including a robot.txt file (also known as the robot exclusion standard) can request the bots to index parts of a website, or nothing at all.

During the early days of internet, the web pages available on the world wide web were very large and even the largest crawlers built during those days were not accurate enough to make indices of those large pages. Therefore search engines were not very optimized those days, whereas nowadays that limitation has been eradicated by modern search engines, which manage to give very accurate results instantly.

Understanding about web crawlers is a very hard task. The pages available today in the internet is different from one another, which indirectly means that the web crawlers that we create should also be different from one another to suit specific tasks.

5.1.1. How Web Crawlers Work

Web crawlers will start with a set of URLs at the beginning, and these URLs are called "Seeds". When the crawler visits these URL's one by one, it will read the contents in those URL's and collect the hyperlinks and will add them to the list of URLs to be visited, and this list is called the "Crawl Frontier". These URLs will be visited by the crawler in a recursive manner according to a set of policies and crawling strategies, whereas, if archiving of websites is needed by the system, it will ensure to save all the visited webpages.

In the process of web crawling, it will face a lot of issues like, large websites, inefficiency, bandwidth and duplicate contents. So at each URL scan, the crawler must carefully choose which pages to visit next.

5.1.2. The Architecture of a Web Crawler

The abstract architecture of a web crawler can be defined in many ways as in Figure Figure5.1. The quality and features of each architecture, depends on the "crawling strategy" being used and the "crawling policies" being used.

5.1.3. Crawling Algorithm

Following pseudo code represents the basic flow of how a web crawler works. We can see different implementations being used in many places. But this will let you get a general idea on how it actually works in terms of implementation. Refer figure Figure5.2.

1. Initialize queue (Q) with a known set of URLs (seed URLs)
2. Loop until Q is empty, or a certain condition is met
3. Pop/Dequeue (Depends on the implementation technique being used) URL (L) from Q
4. If L is not an HTML page (.gif, .jpeg, etc) then exit loop
5. If L is already visited, continue loop (get net URL)
6. Download page (P) for L
7. If cannot download P (404 error, robot.txt included in L), exit loop
8. Else, do the following steps
9. Index P (add to inverted index or store cached copy)
10. Parse P to obtain a list of new links (N)
11. Append N to the end of Q

5.2. Ontology

In computer science, an ontology is a data model that represents knowledge as a set of concepts within a domain, and the relationships between these concepts. When describing an ontology, it normally happen within a certain scope of a domain. Understanding and creating ontologies for the entire universe would be meaningless and endless.

In line with the philosophical definition, an ontology is a formal naming and definition of the types, properties and interrelationships of entities, that really or fundamentally exist for a particular domain of discourse. It is thus a practical application of philosophical ontology, with a taxonomy (classification).

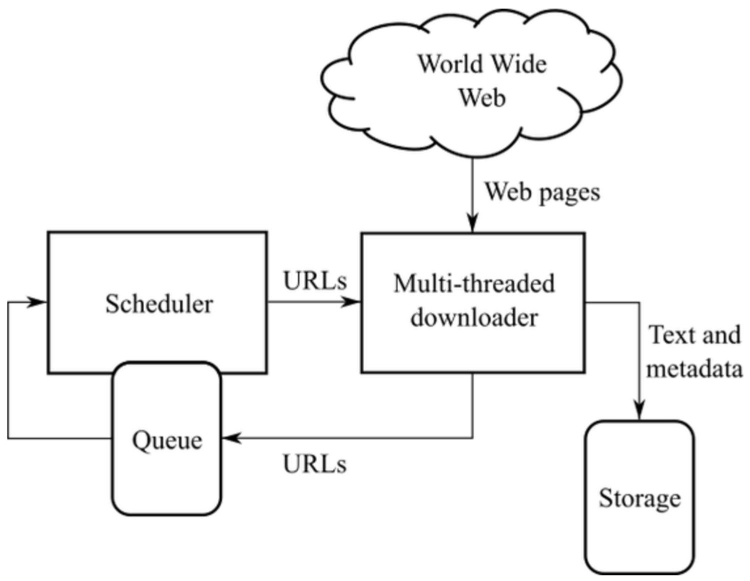


Figure 5.1: Web Crawler Architecture

Here an ontology is the description of what exist specifically within a determined field (domain). Unlike the philosophers mentioned above, these researchers are not primarily interested in discussing if these things are the true essence or the core of the system. They do not even compare what is real between the parts within the system (physical materials) and the processes happening in the system (immaterial concept). The purpose is to understand and describe the underlying structures that affect the individuals and groups.

5.2.1. Uses of Ontologies

Today, people have access to more data from various sources broadening to many different domains and information systems. The amount of data that can be accessed within a single date has increased over time, when compared with the information systems we had decades ago.

For example, if we look at an enterprise, their data sources can be found in many different forms like spreadsheets, databases, presentations, documents, Visio diagrams etc. Since these are all captures in many different formats, it makes it inherently hard to understand the relationship between different data. In a situation like this, it is very hard to understand how policies captured in word documents, relate to business processes captured in models,and how these business processes relate to data captured in the database and so on.

Data needs to be allowed to represent in a format where we can identify all these relationships and stored. Ontologies capture data in a way that allows these relationships to become visible. An ontology is a form of knowledge management. It captures the knowledge within a certain domain (organization/ information system) as a model (data model). This model can then be queried by users, to answer complex questions and display relationships across a domain.

According to the Tom Gruber, an AI specialist at Stanford University, "an ontology is the specification of conceptualizations, used to help programs and humans share knowledge." This means that an ontology is a set of concepts (things), events and relations that are specified using natural language, in order to create an

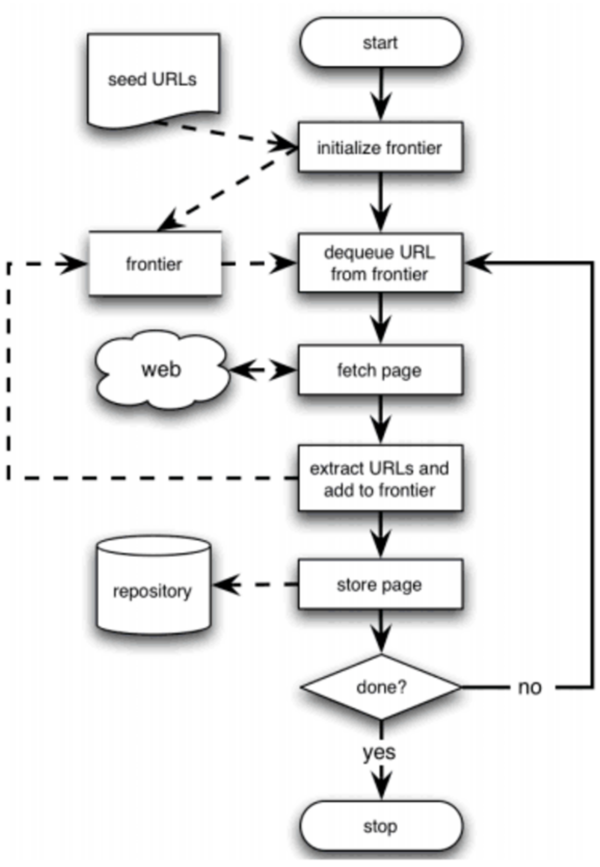


Figure 5.2: Web Crawler Algorithm

agreed-upon vocabulary for exchanging information.

- Conceptualization - Breaking the world into concepts in terms of Entities
- Specification - This is the representation of this conceptualization in a concrete form.

5.2.2. Main Components of an Ontology

A. Concept

A concept represents a set or class of entities or 'things' within a domain. For example, "case law" can be taken as a concept in the legal domain. There are two kinds of concepts.

1. **Primitive Concepts** : These are those which only have necessary conditions (in terms of their properties) for the membership of this class. As an example, for a doctor to practice in Sri Lanka, he/she should necessarily have the SLMC certificate.
2. **Defined Concepts** : These are those whose description is both necessary and sufficient for a thing to be a member of the class. As an example, for a doctor to practice in Sri Lanka, he/she should have the SLMC as well as the Degree.

B. Relations

Relations describe the interactions between concepts or a concept's properties. There are two kinds of relations.

1. Taxonomies - organize concepts into sub- super-concept tree structures
2. Associative - relationships that relate concepts across tree structures

C. Instances

These are the 'things' represented by a concept. For example in the classroom system, "Prof Maurice" is an instance for the concept "Teacher". An ontology does not have any instance. It is merely a conceptualization of the domain. The combination of the ontology with associated instances is what is known as a "knowledge base".

D. Axioms

Axioms are used to constrain values for classes or instances. Like Age should be greater than zero and less than 120.

5.3. Word Embeddings

We always use different mechanisms to represent items in the real world. To represent a scenario, we use articles or movies. To represent a flower, we might use an image. To represent love, we might use music. All of these are different ways in representing certain objects that we use everyday. Similarly, word embedding is a representation of a word. This representation is done using a vector space where each word is represented as a vector. When we consider vectors, they are meaningless alone, but they give semantic to each of them when comparing them within a given vector space. For example, similar words have similar vectors, and vector similarities are more easy to compute. The mapping from word to vectors is simply a mapping of each word to a number (real number). Word embeddings are one of the strongest areas in natural language processing, where vectors whose relative similarities correlate with semantic similarities.

Word embedding is a term given for a set of language modeling and feature learning techniques in Natural Language Processing where words or phrases from the vocabulary are mapped to vectors of real numbers. In simple terms, this is a mapping of a word, to a d-dimensional vector space. Conceptually it involves a mathematical embedding from a space with one dimension per word to a continuous vector space with much lower dimension. We say lower because, unlike other vector representations, word embedding uses a technique to learn lower dimensional (typically between 50 and 1000) dense vectors for words, using the same intuition.

Importance of Word Embeddings

Word embedding is a mapping between words and vectors. In order to understand why we need word embeddings, lets try to identify the main reasons to use vectors instead of the characters in a word for semantic tasks. main two reasons are as follows.

1. Semantic equivalence using characters only is impractical, where vectors do a much better job
2. Computers can handle numbers much better than strings.

When we consider a simple piece of text, a word is considered to be the simplest unit that we can use in identifying semantics of the text. In some simple tasks, we try to use the word and its characters to do the operations. But the word itself, as a sequence of characters is meaningless in terms of semantics. Some words like dog, dogs or eat, eaten or type, typed show a similarity in meaning and representation, and such similarities can be easily understood using the *Levenshtein distance*. But this seems to be impractical in most cases when trying to identify the similarity between words in a given text. (Eg: lawyer and court does not have a similarity in terms of their characters, even though they have a semantic equivalence).

Also, asking a computer to perform operations on strings is very costly and numbers do a better job which makes the computations faster and more efficient.

5.4. Word2vec

Word2vec is a group of related models used to produce word embeddings. It is a two layer neural network, that will process text and train to reconstruct linguistic contexts of words. The word2vec tool takes a text corpus as an input and produces a vector space, typically of 100 - 1000 dimensions, with each unique word in the corpus being added to a vocabulary and assigned a vector in the space. given below in figure Figure5.3 is a very high level look of the word2vec process.

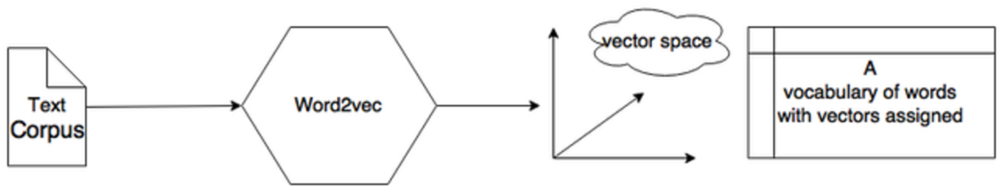


Figure 5.3: Word2Vec Abstract View

Word2vec is not a deep neural network, but the output given by it in numerical form, can be used to process within deep neural networks. The positioning of these vectors (word embeddings) in the vector space occurs in a way such that, the words that share a common context are located in close proximity to one another in the space. The main advantage here is that, this does not need a human intervention at all.

Word2vec is not just a text parsing system. It's applications and extensions are being widely used many fields like, medicine, law, music, movies and many more. Ifwe look deep into its applications, even though word2vec means simply about converting words to vectors, the same process can be applied to other data types like genes, likes, codes, playlists or symbols as well. Words are normal discrete states like the data that was mentioned, and the transitional probabilities between those states (likelihood that they will co-occur) will be searched for. Therefore, similar to word2vec, gene2vec, like2vec and all are also possible.

5.4.1. How it Works

Word2vec takes a text corpus as an input and produces word vectors (word embeddings) as an output. If will first construct a vocabulary of unique words from the text corpus which was used as the training data set, and then learns vector representation of words. The resukting word vector files, can be used in many Natural Language and Machine Learning applications.

When enough data and contexts are being given, it gives a very accurate output about the semantics of words, based on the training data set (text corpus). Those semantics can be used to establish word associations between words or cluster documents and classify them by topic. Those clusters can form the basis of search, sentiment analysis and recommendations in such diverse fields as scientific research, legal discovery, e-commerce and customer relationship management.

Word2vec examines the cosine distance between words and expresses the similarity.

A. Algorithms being Used

A word represented by a number is known as a neural word embedding, which is also the simple transformation being done by word2vec. It vectorizes the words as numbers, so that computers can process natural language with ease.

The image given below is an autoencoder. An autoencoder is an artificial neural networkused for unsupervised learning of efficient codings. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for the purpose of dimensionality reduction. But rather than training against the input words through reconstruction, word2vectrains words against other words that neighbor them in the input corpus.

There are two main learning algorithms in word2vec.

1. Continuous Bag-Of-Words (CBOW)
2. continuous skip-gram

In Continuous Bag Of Words (CBOW) architecture,the model predicts the current word from a window of surrounding context words (neighbouring words). According to the Bag-Of-Words assumption, the order of context words, does not influence the prediction.

In the continuous skip-gram architecture, the model uses the current word to predict the surrounding window of context words. This weighs the nearby context words (neighbouring words) more heavily than distant context words.

According to the author, CBOW is faster while skip gram is slower, but skip gram does a better job for infrequent words. For models with large corpora and high number of dimensions,the skip-gram model yields the highest overall accuracy, and consistently produces the highest accuracy on semantic relationships, as well as yielding the highest syntactic accuracy in most cases. However, the CBOW is less computationally expensive and yields similar accuracy results.

5.4.2. Parameters in word2vec Training

1. Training Algorithm Hierarchical Softmax Algorithm :work
    - s better for infrequent words
    - Negative Sampling Algorithm:works better for frequent words and better with low dimensional vectors
2. Sub-Sampling High frequency words often provide little information. Words with frequency above a certain threshold may be subsampled to increase training speed.
3. Dimensionality
4. Context Window Higher dimensionalities will increase the quality of the word embeddings. When increasing dimensionality, after a certain point, the marginal gain will diminish. The typical dimansionality of vectors in word2vec, is set to be 100 - 1000.The size of the context window determines how many words before and after a given word would be included as context words of the given word.

    CBOW –> Recommended window si

    - ze = 5
    - Skip-gram –> Recommended window size = 10

5.5. Td-Idf

TF-IDF, short for Term Frequency - Inverse Document Frequency, is a text mining technique, that gives a numeric statistic as to how important a word is to a document in a collection or corpus. This is a technique used to categorize documents according to certain words and their importance to the document.

This is somewhat similar to Bag of Words (BoW), where BoW is an algorithm that counts how many times a word appears in a document. If a particular term appears in a document, many times, then there is a possibility of that term being an important word in that document. This is the basic concept of BoW, where the word count allows us to compare and rank documents based on their similarities for applications like search, document classification and text mining. Each of these are modeled into a vector space so as to easily categorize terms and their documents. But the general BoW technique, does not omit the common words that appear in documents and it is also being modeled in the vector space.

But when it comes to TF-IDF, this is also considered as a measure to categorize documents based on the terms that appear in it. But unlike BoW, this does provide a weight for each term, rather than just the count. The TF-IDF value measures the relevance, not frequency.

Why TF-IDF

TF-IDF allows us to score the importance of words in a document, based on how frequently they appear on multiple documents.

1. If the word appears frequently in a document - assign a high score to that word (term frequency - TF)
2. If the word appears in a lot of documents - assign a low score to that word. (inverse document frequency - IDF)

The second point given above is the main reason as to why, common words like "the", "were" are given lower scores, because it appears everywhere. It has no specific unique importance to the relevant document. The TF-IDF value can be associated with weights where search engines often use different variations of TF-IDF weighting mechanisms as a central tool in ranking a document's relevance to a given user query.

This is by far, the best known weighting scheme used in information retrieval. TF-IDF gives the importance of each of terms in a document not only as an isolated term, but also as a term within the entire document collection. This leads to ranking and scoring documents, against a query as well as classification of documents and modeling documents and terms within a vector space.

The *term frequency* is calculated using equation 5.1, where T_d is the most occurring term in document *d*. $TF_{t,d}$ is the frequency of term t in document d and the *inverse document frequency* is calculated using the equation 5.2, where $IDF_{t,D}$ is the inverse document frequency of t in a corpus D. Finally, the $TF\text{–}IDF_{t,d,D}$ value is calculated using equation 5.3.

$$TF_{t,d} = 0.5\,(\frac{term\ t\ count\ in\ d}{count\ of\ term\ T_d}) \tag{5.1}$$

$$IDF_{t,D} = \log\,(\frac{total\ number\ of\ documents\ in\ D}{number\ of\ documents\ with\ t\ in\ it}) \tag{5.2}$$

$$TF\text{–}IDF_{t,d,D} = TF_{t,d} \times IDF_{t,D} \tag{5.3}$$

5.6. Google PageRank Algorithm

According to Google's formal definition from the Google Paper, PageRank is simply a voting platform which considers links to be votes. If there are many links from different websites to "Website A", then "Website A" is considered to be important. This importance is assigned by different weights, where each of the weights are determined by various factors.

This was initially developed by Larry Page and Serge Brin to Rank web pages in Search Engines. This is capable of ranking pages based on the number of links pointing to them. Even at present, this is considered as the basis for all modern search engines. Initially, this is what made Google's Search Results so successful over many other search engines in that day. Although Google was not the first to look at links and create rankings, they were the ones who did it in a meaningful way.

Google PageRank comprises of both its algorithms as well as the score given by the algorithm. PageRank is not the only technique that Google uses to rank its web pages, but the mix of techniques that makes it successful in the web search industry. What we know, and what is being released by Google, regarding the PageRank algorithm is merely a smaller version of it. Google does not clearly mention anything regarding the value that the algorithm outputs. But studies show that this value is a logarithmic scale with a base of about 16.

For example:

- PageRank 2 is 16 times bigger than a PageRank 1
- PageRank 3 is 256 times bigger than a PageRank 1
- PageRank 4 is 4,096 times better than PageRank 1
- PageRank 5 is 65,536 times better than PageRank 1
- PageRank 7 is 4,294,967,296 (over 4 billion) times better than PageRank 1

The PageRank Algorithm

The PageRank algorithm gives a rating of a page's importance in the given context. This ranking is measured recursively, because the importance of one page, refers back to the importance of other pages that link to it. The PageRank algorithm given below, gives a probability distribution as to how likely a certain webpage is, for a person to randomly click on links and arrive at that page. Figure 5.4 depicts a description on the general algorithm being used.

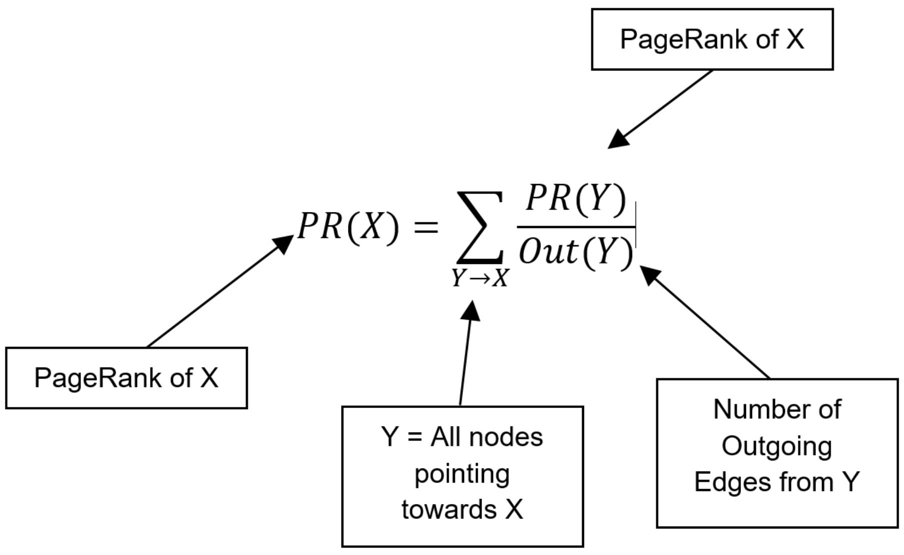

PageRank of X
$PR(X) = \sum_{Y \to X} \frac{PR(Y)}{Out(Y)}$
PageRank of X
Y = All nodes pointing towards X
Number of Outgoing Edges from Y

Figure 5.4: Page Rank Algorithm

5.7. Spell Correction System

When it comes to on-line resources, there is no clear mechanism mentioned for programmers to understand and develop their own spell correction systems. Until 2005, Google had provided a spell checking API, that can easily detect spelling mistakes with few lines of code. It would print out all the corrections available for the input text. But soon after it was shutdown in 2005, Peter Norvig's Spelling Corrector system became very popular, and many variations of it were published. This blog addresses the important concepts behind Norvig's spelling corrector system, and tries to give the reader a mathematical understanding on how it works.

A Probability Model

Statistics and Probability are two powerful tools. What if we can say that the likelihood of something occurring is greater than the other. For example, if we have some statistics that show, what i eat on rainy days and on sunny days. If it is a sunny day, i would eat ice-cream and if it is a rainy day, I would eat a pudding. Based on this information, I have a higher likelihood of eating pudding on a day that is cloudy and dark.

Similarly, if certain information is given (statistics), we can give a likelihood (probability) of a spell correction for a given word. Before moving on, let's get familiar with the notation I will be using in this section.

- w = The original word (this is the word that needs to be checked)
- c = The candidate for the spell correction of word w.

The problem with probabilistic models is that, we can never know for sure, the correct spelling correction needed for a given word. It is always about the most likelihood that we consider in most of these cases. So, we are supposed to pick the correction (c) out of all possible candidates, which would maximize the probability.

Hope you are familiar with the Bayes' Theorem. If we apply the Bayes' Theorem to this scenario, we need to figure out how to get the maximum out of P (c|w), which means, the probability of C being the correction term, given w. The probability model is depicted in Figure 5.5

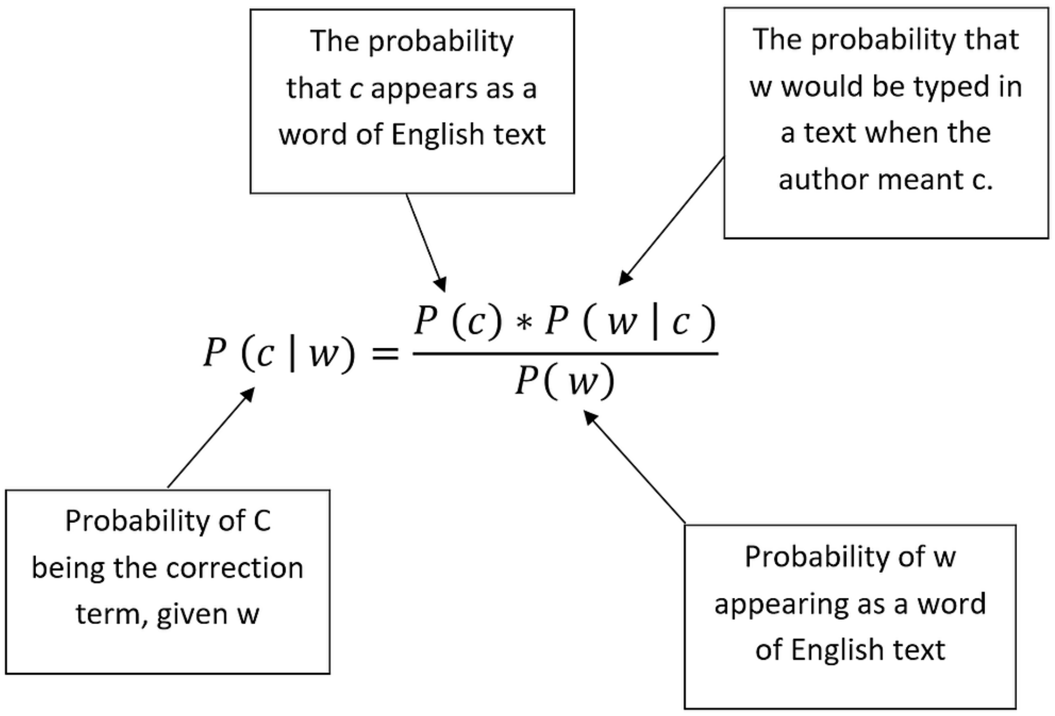

The probability that *c* appears as a word of English text
The probability that w would be typed in a text when the author meant c.
$P(c \mid w) = \dfrac{P(c) * P(w \mid c)}{P(w)}$
Probability of C being the correction term, given w
Probability of w appearing as a word of English text

Figure 5.5: Spell Correction Probability Model

6 Methodology

The proposed solution is based on the following simple principle: No more painful querying. Therefore, the proposed solution is an ontology based legal information extraction system for the United States legal system, which will guide lawyers and paralegals to get relevant legal information regarding a court case of interest, with both keyword and natural language based query processing.

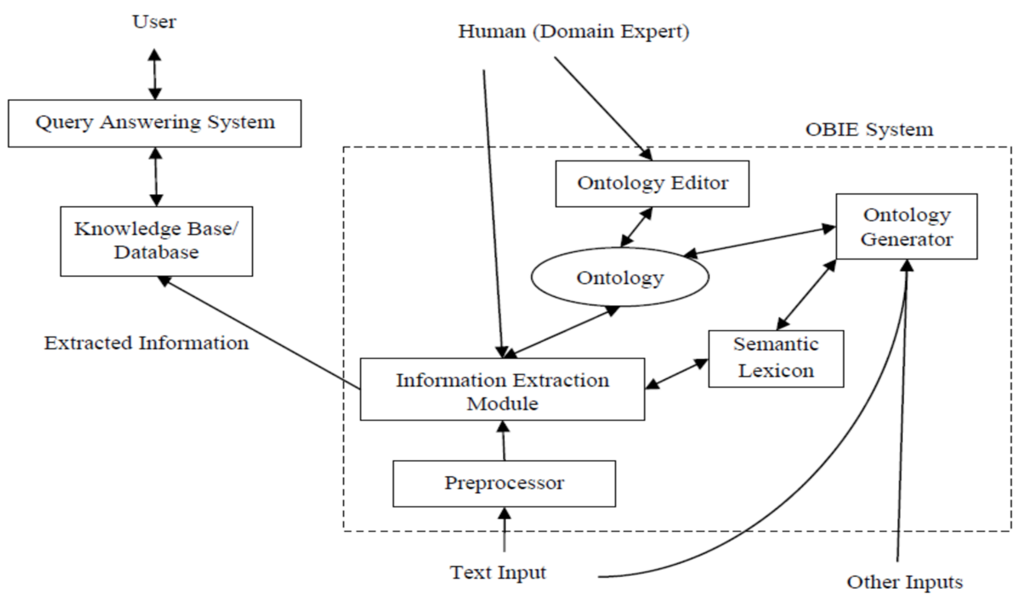


Figure 6.1: Ontology Based Information Extraction Architecture

The architecture of this system is based on the general architecture of an Ontology Based Information Extraction (OBIE) system [1], as depicted in Figure 6.1. Firstly, the relevant information required on court cases, judgments and laws of the United States legal system, will be collected using a web crawler from the FindLaw website. Having all the legal documents of the US legal system available online is the main reason for using the US legal system, where direct effort can be given to computations, in contrast to other legal systems, where most effort will be given to digitizing those legal documents. Once the required information has been gathered using web crawlers, it will be stored in a database. For this purpose, the system will be using a relational database model like MySQL. The collected data will be sent through a preprocessor, in order to eliminate the redundancies associated with the collected data and make it possible for the NLP systems to handle the data. This is merely a filtering process, where it looks for redundancy and irrelevant data types, which are being gathered into the databases using the web crawlers. The preprocessor will also run a spelling and grammar checker on the text corpuses to ensure the consistency of the data being collected.

Next, the ontology generator will create a law ontology, using the data collected from the above process. This process will also be guided by the legal domain experts, to ensure the accuracy of developed system. Once the ontology is generated, it will be manually edited (if required) for fine tuning using an ontology editor like Protégé [1]. With this, the ontology generation process will become semi-automatic [1], in order to increase the accuracy and the reliability of the ontology being generated. The generated law ontology will be stored using a NOSQL database like Cassandra. “The reason for using Cassandra is the promising results and feedbacks from ontology systems which adapted the initial query language under Cassandra environment, with exploitation of massive data and their distributions” [162].

Afterwards, the information extraction process will be driven by the generated law ontology as it will be used to derive the linguistic rules and gazetteer lists to extract necessary information. The natural text for information extraction will be fed into the Natural Language Processing (NLP) pipeline for the purpose of parts of speech tagging, name entity recognition and pre-processing. According to the general Ontology Based Information Extraction (OBIE) architecture [1], the information extraction process will also be supported by a semantic lexicon, which is a dictionary of words, labeled with semantic classes both in legal domain and general domain. Due to this component, associations can be drawn between words. In a very reductionist view, it is a dictionary with a semantic network.

After the extraction process, the extracted information will be stored in a knowledge base and the relevant legal information for the given query (previous court case, judgments, and laws) or request, will be provided to the user as the output. This unit is named as query answering module, which provides answers to the user queries submitted in both natural text and syntactic queries. A web application will be provided to the user to enter queries or natural text related to the legal domain, and the query answering module will retrieve the relevant information from the knowledge base created above.

This section describes the overall methodology required in this study to build a proper Ontology Based Legal Information Extraction System. A proposed high level course of actions is given below, where each of them are briefly being described as given below. The general architecture of an Ontology Based Information Extraction system is shown in figure Figure6.1 where we have used an extension of this architecture to support and cater to the legal domain.

This Chapter will first describe the core system being developed in this study, and explain each segment in depth, so that the reader has a clear idea on what has been done and what tools have been used for this process. As mentioned in Chapter 1, this system consists of the following main sub-components, which will be addressed within this chapter. In the next Chapter, we will be talking about the experimental results of this study and how favourable each result has been, for our study.

- Data Collection
- Preprocessor
- Semantic Similarity
- Legal Ontology
- Ontology Class Vectors Generation
- Semi-supervised Ontology Instance Population
- Information Retrieval Module
- Linguistic Rules and Gazetteer List
- Information Extraction Module
- Query Answering Module

The diagram in Figure 6.3 depicts the basic idea behind the overall methodology of the system. This overall diagram will be explained in terms of subcomponents of the main system.

[1] http://protege.stanford.edu

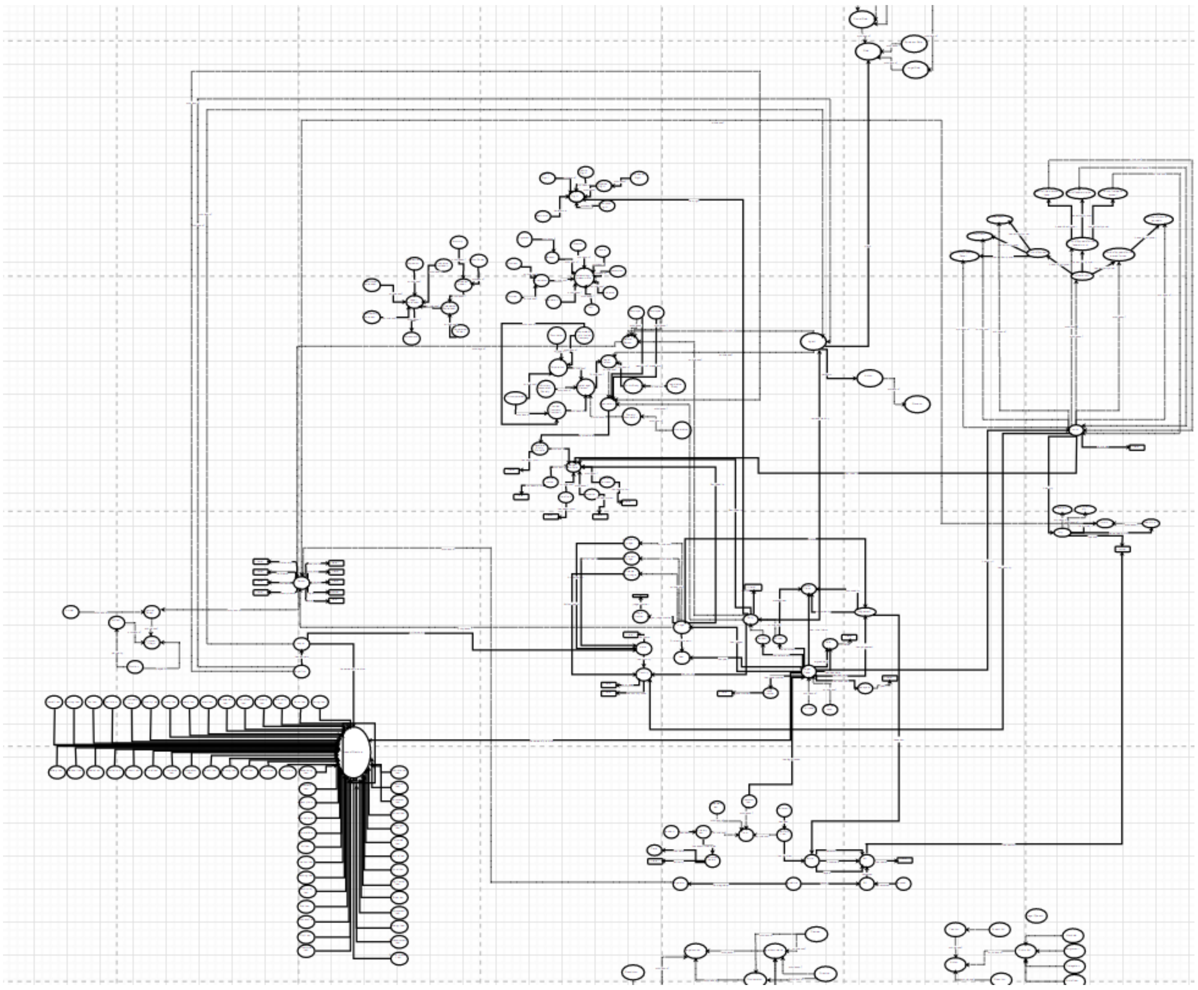


Figure 6.2: Legal Ontology Knowledge Map

6.1. Data Collection

This section will address the primary part of the entire process that explains the data collection process. This is needed to initiate the entire process where the whole system is dependent on the primary fact of data collection. This section will address components like how the data is structures, where to locate it, the crawling aspect to gather the required data repositories,and so on.

6.1.1. About the Data

In this study, data collection was carried out using web crawling on several online legal repositories like, Findlaw [2], which has a huge legal case databases, that gets updated regularly, everyday. These repositories contain the United States Law system, which is our primary focus and scope of this study. The main reason as to why we use the United States Legal System has plenty of reason. The primary reason is that, the legal information being available online where the effort to digitize the legal data is reduced. In certain countries like Sri Lanka, Bangaladesh, the legal systems solely run on manual intervention and local document repositories such as book, journals, libraries and so on. The main drawback in these systems is the effort that has to to be included in trying to find the relevant legal cases needed, whenever there is a requirement.

By adding our fullest focus on using the online repositories available, we mitigate the additional effort needed in digitizing this data, whereas our fullest focus can be incorporated into the computing aspects of information retrieval and extraction.

As in many other legal systems, the United States Legal system follows a structured format when reporting

[2] http://www.findlaw.com/

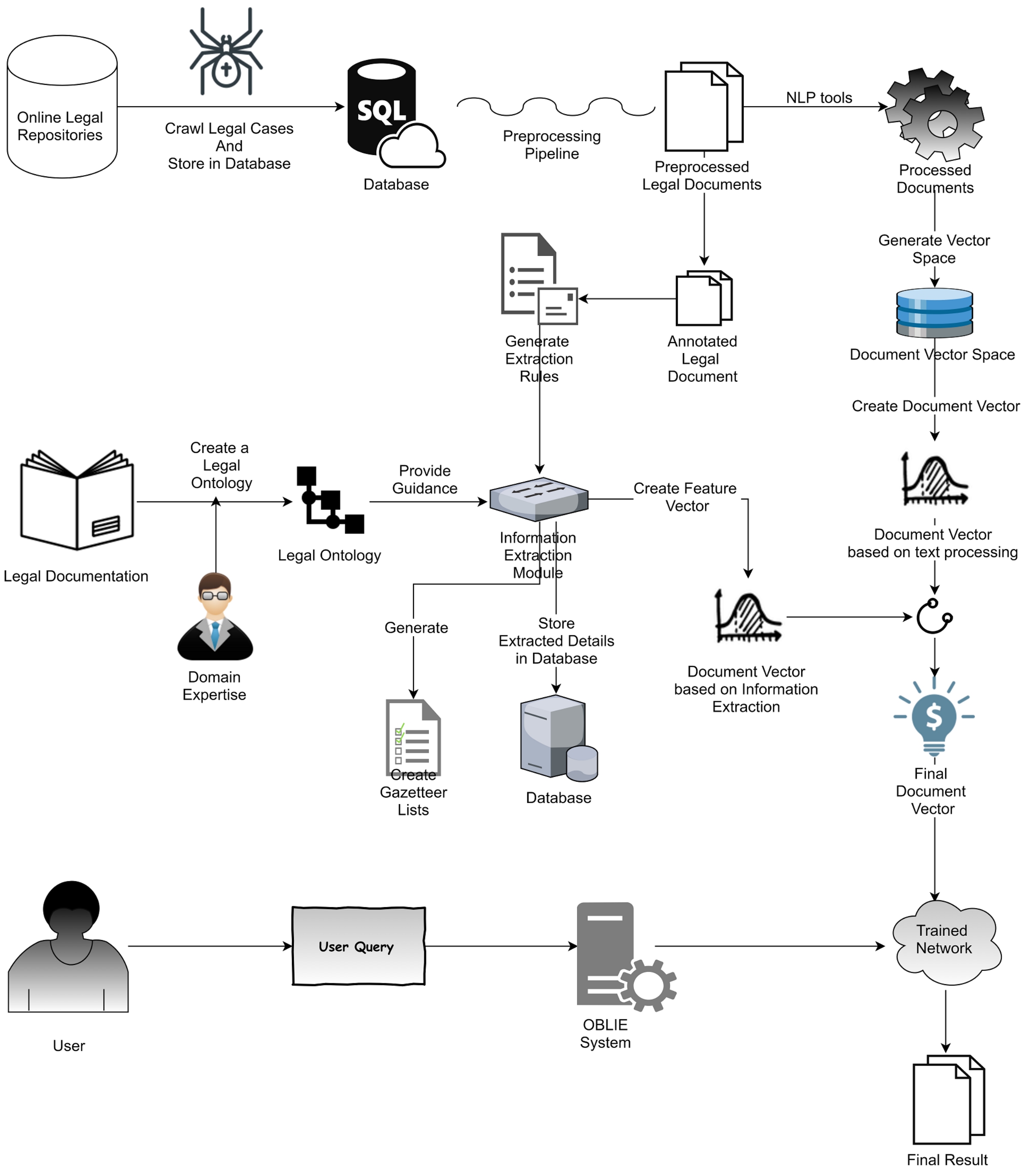


Figure 6.3: Overall Methodology of the System

legal cases online, which makes it easier for us to retrieve relevant documents needed. Hence, it becomes easy for us in terms of scalability of the database or the repository, where the system we build, can easily be extended into any other legal domain system available online.

6.1.2. Web Crawling

A web crawler is a software bot (internet bot) that will surf through the world wide web in a systematic manner and collects required information in a meaningful way. Typically it's purpose is to do web indexing (web spidering). For Web Crawling purposes, we used two libraries specifically created for this purpose, namely BeautifulSoup and JSoup. BeautifulSoup is a python library whereas the latter is a Java library. Due to huge amounts of data being gathered in the web crawling process, and the capability of Java to better handle huge data and parse them according to our information needs using the JSoup library, give us the upper hand to go forth with using Java as the main platform for web crawler implementation.

There is always no hard and fast rules to implement a Web Crawler. The toughest part in implementing a web crawler is to study the structure of a website and how its data is being structured in the relevant blocks of a web page. This process is very time consuming, and needs to be done in a careful manner. Failing to do so, might cause the process to fail in between by failing to retrieve relevant hyper links and information as expected.

The process of web crawling was very time intensive, where it took over 200 hours of computational power to crawl around 30,000 legal cases from the Findlaw online repository. But in the latter part of this study, we had to recrawl for certain information from the same repositories, where over 2500 legal cases were retrieved within a time span of one and a half weeks f computation power. We collected, legal cases, judgments, statutes and laws.

6.2. Preprocessor

The legal documents collected using web crawling requires additional processing due to its syntactical and semantic meanings in the text. This is a common issue arising in domain specific documents that needs adherence where special types of preprocessing and text mining techniques need to be included further into the process, to get it to a working condition. But the main problem arising is the fact that there is no one common method to do this preprocessing as it requires domain knowledge and domain specificity, where understanding the domain specific documents is a must to understand what kind of preprocessing techniques should be applied to the text.

In this study, we have applied a set of preprocessing techniques, that has taken the legal document from raw text to usable and machine readable text, where it has been further used in the study. This is depicted in Figure 6.4.

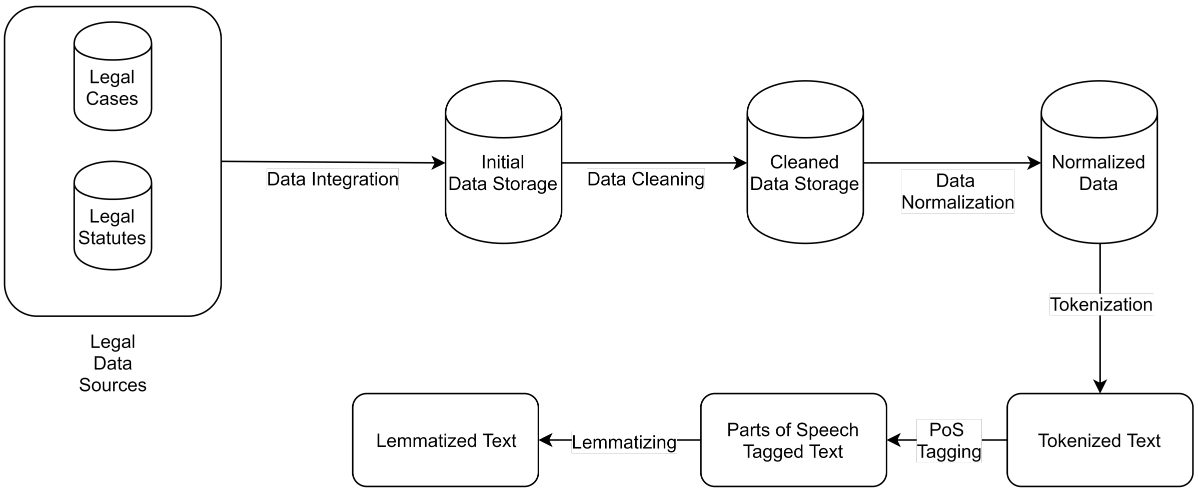


Figure 6.4: Data Preprocessing on Legal Documents

6.3. Semantic Similarity

This section explains the idea of developing a system that can identify the similarity measures of legal domain words. This method proposed in this section is a direct link to the upcoming methodologies, where the process of identifying semantic similarity between legal documents is also supported by this process.

An overview of the methodology of this subtopic we propose is illustrated in Figure 6.11. The first phase of this study was to gather the necessary legal cases from on-line repositories. We obtained over 2500 legal case documents, pertaining to various areas of practices in law, from Findlaw[3] by web crawling. Due to this reason, the system tends to be generalized well over many aspects of law.

[3]http://caselaw.findlaw.com/

6.3.1. Text Lemmatization

The linguistic process of mapping inflected forms of a word to the word's core lemma is called lemmatization [163]. The crawled natural language text would contain words of all inflected forms. However, the default word2vec model does not run a lemmatizer before the word vector embeddings are calculated; which results in each of the inflected forms of a single word ending up with a separate embedding vector. This in turn leads to many drawbacks and inefficiencies. Maintaining a separate vector for each inflected form of each word makes the model bloat up an consume memory unnecessarily. This especially leads to problems in building the model. Further, having separate vector for each inflected form weakens the model because the values for words originating from the same lemma will be distributed over those multiple vectors. For example, when we search for similar words to the noun input "judge", we get the following words as similar words: *Judge, judges, Judges.* Similarly, for verb input "train", the model would return the following set of words: *training, trains, trained.*

As shown in Figure 6.11, we use the raw legal text to train the $word2vec_{LR}$ model. But for both $word2vec_{LL}$ model and $word2vec_{LLS}$ model, we first lemmatize the legal document corpus to map all inflected forms of the words to their respective lemmas. For this task we use the Stanford CoreNLP library [103].

6.3.2. Training word2vec models

In this stage we trained wod2vec models. One was on the raw legal text corpus and the other was on the lemmatized law text corpus. Each input legal text corpus included text from over 2500 legal case documents amounting to 20 billion words in all. The training took over 2 days. As mentioned in the Section **??**, the model trained by the raw legal text corpus is the $word2vec_{LR}$model shown in Figure 6.11. The model trained on lemmatized law text corpus is $word2vec_{LL}$. Further, a clone of the trained $word2vec_{LL}$is passed forward to **??** in order to build the $word2vec_{LLS}$model. Following are the important parameters we specified in training these models: size (dimensionality) set to 200, context window size set to 10, learning model is CBOW, min-count is set to 5, training algorithm is hierarchical softmax.

For this phase, we used a neural network with a single hidden layer as per the experiments by Mikolov [9]. As mentioned above, to train the system to output a user-specified dimensional vector space, we picked the Continuous Bag Of Words approach for learning weights. The rationale as to why CBOW learning model was picked over skip-gram is that, Skip-gram is said to be accurate for infrequent words whereas CBOW is faster by a factor of window size which is also more appropriate for larger text corpora.

In CBOW, the current word or the target word is being predicted based on the context words, within the specified context window size.

In addition to the above trained models, we obtained the $word2vec_{G}$model which was trained by Google on the Google News dataset. As shown in Figure 6.11, this is a generic text corpus that contain data pertaining to a large number of topics. It is not fine tuned to the legal domain as the three models ($word2vec_{LR}$, $word2vec_{LL}$, and $word2vec_{LLS}$) that we train. However, Google's model was trained over on around 100 billion words that add up to 3,000,000 unique phrases. This model was built with layer size of 300. Due to the general nature and the massive case of $word2vec_{G}$model, it is possible to use the comparison of results obtained by our models against the results obtained by this model to showcase the effectiveness of a model trained using a specific domain in the applications of that domain over a model trained using a generic domain in application on the same specific domain.

A. Mathematical model

The first step was to obtain an entry from the training set and to query the $word2vec_{LL}$model with the key lemma. Let us define an integer n such that $n = Cl$ where $C > 1$. When executing the query, $word2vec_{LL}$model was instructed to return the first n elements that matches the query best along with the word2vec similarity values. Let us call this resultant Matrix R. R has n columns where w_i is the word similar to k and d_i is the word2vec similarity between k and w_i. Equation 6.1 shows R.

$$R = \begin{bmatrix} w_1 & w_2 & w_3 & \dots & w_n \\ d_1 & d_2 & d_3 & \dots & d_n \end{bmatrix} \tag{6.1}$$

From R, we created the vectors W and D as shown in Equation 6.2 and Equation 6.3 respectively. Each w_i and d_i has the same value as they had in R.

$$W = \{w_1, w_2, w_3, ..., w_n\} \tag{6.2}$$

$$D = \{d_1, d_2, d_3, ..., d_n\} \tag{6.3}$$

We defined the following functions for words w_i and w_j. The lexical semantic similarity calculated between w_i and w_j using the Wu & Palmer's model [15] was given by $wup(w_i, w_j)$. The lexical semantic similarity calculated by Jiang & Conrath's model [16] was given by $jcn(w_i, w_j)$. $hso(w_i, w_j)$ was used to indicate the lexical semantic similarity calculated using the Hirst & St-Onge's system [17].

Next we created the lexical semantic similarity matrix M. The matrix M is a $4 \times N$ matrix where N has the same value as in Equation 6.1. The first row element $m_{1,i}$ of M was calculated by taking the Wu & Palmer similarity between k and w_i. Here w_i element at index i of W. Thus, $m_{1,i}$ is equal to $wup(k, w_i)$. Similarly, the second row element $m_{2,i}$ of M was calculated by taking the Jiang & Conrath similarity between k and w_i. The third row consists of Hirst & St-Onge similarities while the forth row contains a series of 1s for the sake of the bias. The matrix M is shown in Equation 6.4.

$$M = \begin{bmatrix} wup(k, w_1) & wup(k, w_2) & \dots & wup(k, w_n) \\ jcn(k, w_1) & jcn(k, w_2) & \dots & jcn(k, w_n) \\ hso(k, w_1) & hso(k, w_2) & \dots & hso(k, w_n) \\ 1 & 1 & \dots & 1 \end{bmatrix} \tag{6.4}$$

We defined the normalizing function given in Equation 6.5 to return a value x_{norm} when given a value x_{raw} that exists between the maximum value v_{max} and the minimum value v_{min}. The returned x_{norm} is a double value that has the range $[0, 1]$.

$$x_{norm} = normalize(x_{raw}, v_{min}, v_{max}) \tag{6.5}$$

We created the matrix SM from the matrix M by calculating each $sm_{i,j}$ using the Equation 6.5 on each $m_{i,j}$. For this we used the min and max values shown in table 6.1.

Table 6.1: Lexical Semantic Similarity Statistics

Model	Min	Max
Wu & Palmer	0.0	1.0
Jiang & Conrath	0.0	∞
Hirst & St-Onge	0.0	16.0

We defined the value matrix V using the vector D and the Matrix SM as shown by Equation 6.6.

$$V = \begin{bmatrix} D \\ SM \end{bmatrix}^T \tag{6.6}$$

We defined the vector E where each element e_i is given by activating a neural network by values v_i, where v_i is the ith row of V. The training of the aforementioned neural network is explained in Section 6.3.2.B.

B. Machine Learning for weight calculation

The motive behind using machine learning is to train the system to take into account how each similarity measure value can be used to derive a new, compound, and better representative value for similarity. As mentioned in Section 6.3.2.A, a neural network was chosen as the machine learning method. Initially the weight values were initialized to random values and E was calculated. Next the matrix MI was defined as shown in Equation 6.7.

$$MI = \begin{bmatrix} W \\ E \end{bmatrix} \tag{6.7}$$

The matrix Y was obtained by sorting the columns of matrix MI in the descending order of elements in the second row. We defined a seeking function given in Equation 6.8 to return the index of word w in the first row of matrix P. If the word w does not exist in the first row of matrix P, it returns the column count of the matrix P. Also, observe that the new matrix Y has the same form as the initial matrix R shown in Equation 6.1. This symmetrical representation is important because it gives us the opportunity to use the same accuracy measures on all the word2vec models in Section **??** to achieve a fair comparison.

$$s = seek(w, P) \tag{6.8}$$

Next we defined the error err according to Equation 6.10 where the value of x_i was derived from Equation 6.9. ϵ is a small constant and G is the Golden Standard Vector.

$$x_i = \begin{cases} 1, & \text{if } seek(g_i, Y) < l \\ \frac{n - seek(g_i, Y)}{n - l}, & \text{otherwise} \end{cases} \tag{6.9}$$

$$err = 1 - \frac{\epsilon + \sum_{i=1}^{l} x_i}{|G \bigcap W| + \epsilon} \tag{6.10}$$

This error err is used to adjust the neural network. This training cycle is continued until convergence. The completed model that is trained this way is named $word2vec_{LLS}$model.

6.3.3. Query processing

In order to use and test our models, we built a query processing system. A user can enter a query in the legal domain using the provided interface. The system then takes the query and applies natural language processing techniques such as PoS tagging until the query is through the NLP pipeline to be lemmatized. We used the same Stanford Core NLP pipeline that we used in Section **??** for this task. The reason for this is to bring all the models to the same level to be compared equally. We have shown this step in Figure 6.11.

6.4. Legal Ontology

In computer science, an ontology is a data model that represents knowledge as a set of concepts within a domain, and the relationships between these concepts. When describing an ontology, it normally happen within a certain scope of a domain. The most time consuming part of this project was to generate an ontology for the legal domain. By developing an ontology in this domain, we expect to get a shared conceptualization of the domain, so that it will guide us in the information extraction process to extract relevant information as necessary.

This was created with the help of legal domain specialists and after each iteration of th development process, this was validated and evaluated by the legal professional, to see whether we have matched and identified the relevant concepts and relationships in the legal domain. At first, we initiated this process by targeting the consumer protection law, whereas later with its generality, it was scoped out into other law categories including criminal law. A snapshot of the ontology that was created, is shown in figure Figure6.2.

The ontology generation process was entirely a manual process, where we have currently come up with an ontology to map most of the legal cases available in the text corpus extracted via web crawling. The validation of the level of completeness of the ontology was mapped using manual mapping between legal cases concepts and the concepts being modeled in the ontology. This was done with the help of domain experts, where this was carried out at every iteration of the ontology development process.

The database diagrams are available in the System Requirement Specification Document provided alongside this report.

Visualization and Editing

For ontology visualization and editing we have used the Protege editor where it supports generation of OWL files, and visualization of them. By using this tool, it helps us to look at the generated ontology and then edit the concepts and relationships accordingly. A visualization of the legal ontology is depicted in figureFigure6.5.

6.5. Ontology Class Vector Generation

This section describes the generation of vectors for ontology class names, based on ontology class instances collected. This is later being used in the study to create a fully fledged vector space to support the entire system.

We discuss the methodology that we used for deriving a vector representation for ontology classes using instance vector embeddings in this section. Each of the following subsections describe a step of our process. An overview of the methodology we propose in Sections 6.6.1 and 6.6.2 is illustrated in Figure 6.6 and an overview of the methodology we propose from Section 6.5.3 to Section 6.5.7 is illustrated in Figure 6.7.

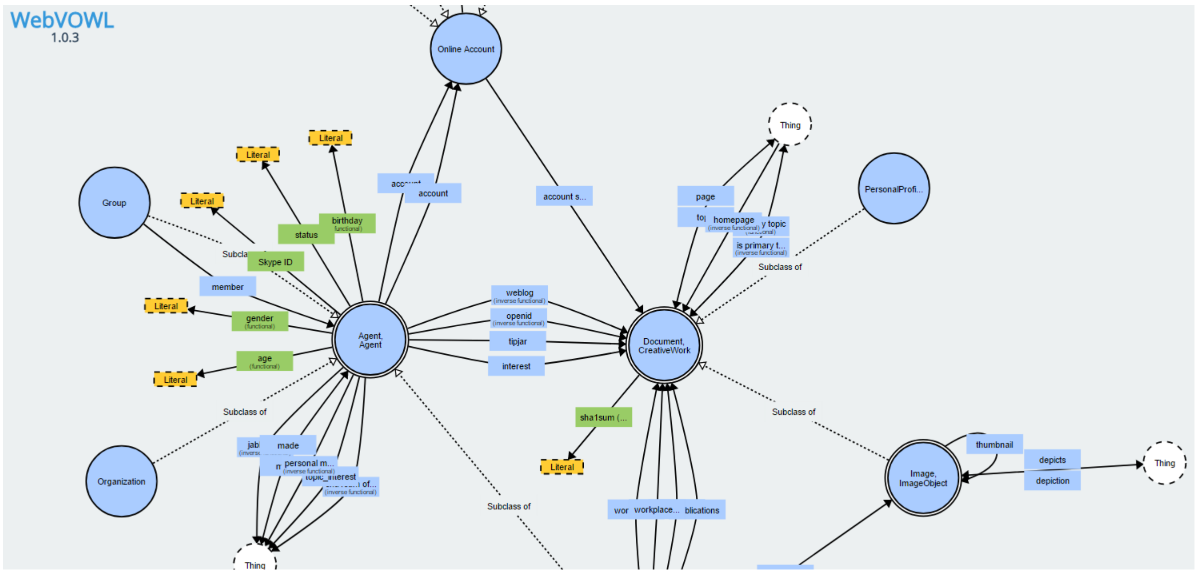


Figure 6.5: Legal Ontology Visualization

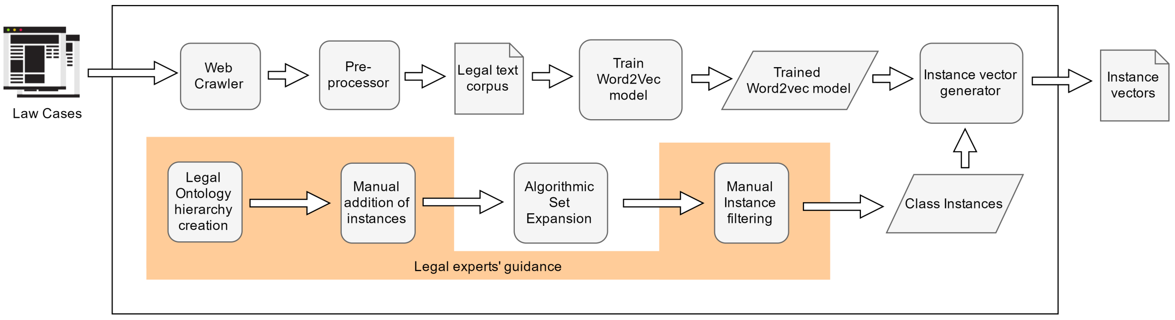


Figure 6.6: Flow diagram of the methodology for deriving instance vectors

6.5.1. Ontology Creation

We created a legal ontology based on the consumer protection law, taking Findlaw[4] as the reference. After creating the ontology class hierarchy, we manually added seed instances for all the classes in the ontology. This was done based on manual inspection of the content of legal cases under consumer protection law in Findlaw. Next we used the algorithm proposed in [20] to expand the instance sets. The expanded lists were then pruned manually to prevent conceptual drift. This entire process was done under the supervision and guidance of experts from the legal domain.

6.5.2. Training word Embeddings

The training of the word embeddings was the process of building a *word2vec* model using a large legal text corpus obtained from Findlaw[4]. The text corpus consisted of legal cases under 78 law categories. In creating the legal text corpus we used *Stanford CoreNLP* for preprocessing the text with tokenizing, sentence splitting, Part of Speech (PoS) tagging, and lemmatizing.

The motive behind using a pipeline that pre-processes text up to and including lemmatization instead of the traditional approach of training the word2vec model with just tokenized text[9], was to map all inflected forms of a given lemma to a single entity. In the traditional approach each inflected form of a lemma gets trained as a separate vector. This dilutes the values that is extracted from the context of that lemma between all inflected forms. Thus sometimes resulting in values not meeting the threshold when a test of significance is done. By having all inflicted forms to be reduced to the relevant lemma and train, we made sure that all the contributions of the context of a given lemma is collected at a single vector, thus making the system more

[4]http://caselaw.findlaw.com/

accurate. Secondly, having a unique vector for each inflected form makes the model unnecessarily large and heavy. This results in difficulties at both training and testing. Our approach of lemmatizing the corpus first solves that problem as well. In addition to this, to reduce ambiguities caused by the case of the strings, we converted all strings to lowercase before training the word2vec model. This too is a divergence from the conventional approach.

6.5.3. Sub-Cluster Creation

By definition, the instances in a given class of an ontology is more semantically similar to each other than instances in other classes. But no matter how coherent a set of items is, as long as that set contains more than one element, it is possible to create non-empty sub-sets that are proper subsets of the original set. This is the main motivation behind this step of our methodology. Following this rationale, it was decided that it is possible to find at least one main schism in the otherwise semantically coherent set of more than one instances.

Given that even then, the schisms would be fairly small by definition in comparison to the difference of instances in one class and the instances of another class, it was decided to stop the sub-set creation at 2. Which means we decided to divide the instances in a single class in to two sub-clusters. For this purpose, we use K-means clustering with K=2. For an example, we are subdividing the "Judge" class using k-means to "Judge1" and "Judge2" and then use the support vectors between those two to predict the label of "Judge". The centers of these sub-clusters are two of the candidate vectors used in Section 6.6.5.

6.5.4. Support Vector Calculation

It is clear that the specification of the problem handled in this study is more close to a clustering task than a classification task. In addition to that, given the fact that each time we would be running the proposed algorithm, it would be for instances in a single class. That implies that, even if we are to assign class labels to the instances, to model it as a classification task, it would have been a futile effort because there exist only one class. Thus, having a classifying algorithm such as Support Vector Machines (SVM) as part of the process of this study might seem peculiar at the beginning. However, this problem is no longer an issue due to the steps that were taken in the Section 6.5.3. Instead of the single unified cluster with the homogeneous class label, the previous step yielded two sub-clusters with two unique labels. Given that the whole premise of creating sub-clusters was based on the argument that there exists a schism in the individual vectors in the class, it is logical to have the next step to quantify that schism.

For this task we used an SVM. The SVM was given the individuals in the two sub-clusters as the two classes and was directed to discover the support vectors. This process found the support vectors to be on either side of the schism that was discussed in Section 6.5.3.

In identifying the support vectors, we used the algorithm used by Edgar Osuna et al. [164] in training support vector machines and then performed certain modifications to output the support vectors.

6.5.5. Candidate Matrix Calculation

With the above steps completed, in order to derive the vector representation for ontology classes, we calculated a number of candidate vectors for each class and then derived the candidate matrix from those candidate vectors. We describe the said candidate vectors below:

- Average support vector (C_1)
- Average instance vector (C_2)
- Class Median vector (C_3)
- Sub-cluster average vectors (C_4, C_5)

The **Class name Vector** (C_0) is obtained by performing word vectorization on the selected class's class name and it was used as our desired output.

A. Average support vector (C_1)

We identified the support vectors which mark the space around the hyperplane that divides the class into the two subclasses as mentioned in Section 6.5.4. We take the average of the said support vectors as the first candidate vector. The rationale behind this idea is that as described in Section 6.5.3, there exists a schism in between the two sub-classes. The support vectors mark the edges of that schism which means the average of the support vectors fall on the center of the said schism. Given that the schism is a representative property of the class, it is rational to consider this average support vector that falls in the middle of it as representative of the class. We averaged the instance vectors as shown in equation 6.31 to calculate the average support vector.

$$C = \frac{\sum_{i=1}^{N} a_i V_i}{\sum_{i=1}^{N} a_i} \tag{6.11}$$

Here, N is the total number of vectors in the class. V_i represents instance vectors. a_i is the support vector membership vector such that: if the ith vector is a support vector, a_i is 1 and otherwise, it is 0. Here C is C_1 which represents the average support vector candidate vector.

B. Average instance vector (C_2)

This is by far the most commonly used class representation vector in other studies. We took all the instance vectors of the relevant class, and averaged them to come up with this candidate vector for the class. The Average instance vector was also calculated using the equation 6.31. However, this time all the a_is were initiated to 1. In that case, C is C_2 which represents the average instance vector.

C. Class Median vector (C_3)

Out of the instance vectors of a class, we calculated the median vector of them and added it as a candidate vector for that class. The Class Median vector (C_3) was calculated as shown in equation 6.15 where: V is the set of instance vectors in the class. v_i is the ith instance vector. M is the number of dimensions in an instance vector. $x_{avg,j}$ is the jth element in the average instance vector C_2 that was calculated above using equation 6.31.

$$C_3 = arg \min_{v_i \in V} \left\{ \sum_{j=1}^{M} (x_{i,j} - x_{avg,j})^2 \right\} \tag{6.12}$$

D. Sub-cluster average vectors (C_4, C_5)

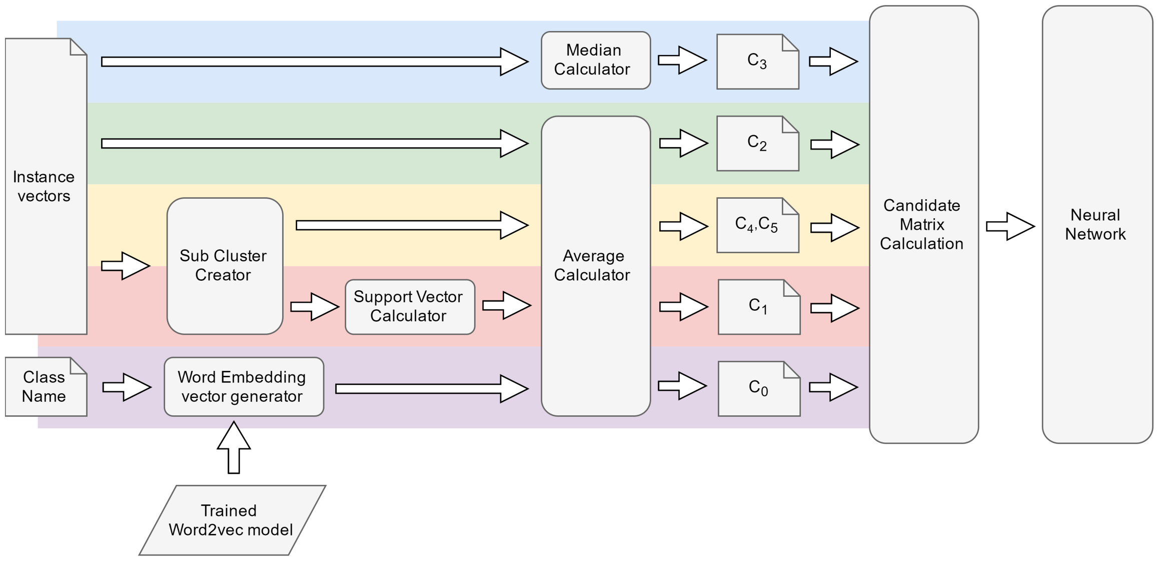


Figure 6.7: Flow diagram of the methodology for training and testing the neural network

We took all the instance vectors of one sub-cluster and averaged them to calculate C_4 and then did the same for the other sub-cluster to calculate C_5. The rationale behind this step is the fact that as described in Section 6.5.3, the two sub-clusters that we find are representative of the main division that exists in the class. Thus it is justifiable to consider the centroid of each of those sub-clusters.

Each sub-cluster average instance vector was also calculated using the equation 6.31. However, this time all the a_is in the first cluster was initiated to 1 and the a_is in the second cluster was initiated to 0. In that case, C is C_4 which represents the average instance vector for the first cluster. Next the same function was used with all the a_is in the first cluster initiated to 0 and the a_is in the second cluster initiated to 1 to calculate C which was assigned to C_5.

E. Candidate Matrix

Finally the candidate matrix M_{cand} for each class is made by concatenating the transposes of C_1 though C_5 as shown in equation 6.18.

$$M_{cand} = \left\{C_1^T, C_2^T, C_3^T, C_4^T, C_5^T\right\} \tag{6.13}$$

6.5.6. Optimization Goal

After calculating the candidate vectors, we proposed an optimal vector that represents the given class based on the optimization goal as follows:

$$Y = \frac{\sum_{i=1}^{M} C_i W_i}{\sum_{i=1}^{M} W_i} \tag{6.14}$$

Here, Y is the predicted class vector for the given class. M is the number of candidate vectors for a given class. C_i and W_i represents the ith candidate vector and the associated weight of that candidate vector respectively. Here the W_i is calculated using the method described in Section 6.5.7.

6.5.7. Machine Learning implementation for weight calculation

The main motive behind adding a weight for each candidate vector is to account for the significance of the candidate vector towards the optimized class vector. We decided to use machine learning to calculate the weight vector. The machine learning method we used is a neural network.

The dataset is structured in the traditional (X, D) structure where X is the set of inputs and D is the set of outputs. An input tuple x (such that $x \in X$), has elements $x_{i,j}$. The matching output tuple d (such that $d \in D$) has a single element.

X and D is populated as follows: Take the ith row of the candidate matrix M_{cand} of the class as x and add it to X. Take the ith element of C_0 of the class and add it to D. Once this is done for all the classes, we get the (X, D) set. It should be noted that since the weights are learned universally and not on a class by class basis, there will be one large dataset and not a number of small datasets made up of one per class. The reason for this is to make sure that the model does not overfit to one class and would instead generalize over all the data across the entire ontology. Because of this approach, we end up with a considerable amount of training data which again justifies our decision to use machine learning. For a word embedding vector length of N over M number of classes, this approach creates NM number of training examples.

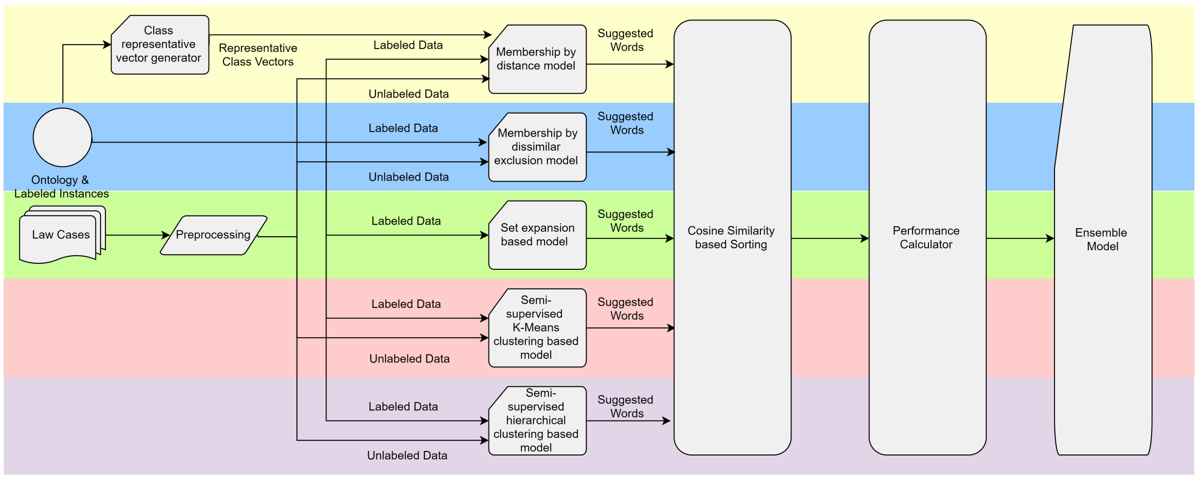
6.6. Semi-supervised Ontology Instance Population

This section explains the concept of ontology population using semi-supervised machine learning techniques, that help collect the necessary instances for Ontology classes from legal text documents. This study is further used in this research for our future endeavours.

We discuss the methodology used in this research study in the upcoming sections. Each of the following subsections describe a step of our process. An overview of the methodology we propose is illustrated in Figure 6.8.

6.6.1. Ontology Creation

For the ontology creation, we focused on the consumer protection law domain and created a legal ontology, based on Findlaw [165] as the reference. However, for the sake of clarity of this paper, we extract a sub-ontology from it and use it to explain the methodology to make the process simple and intuitive to understand. In selecting a part of the ontology, we mainly focused on more sophisticated relationships and taxonomic presences. An overview of the selected part of ontology is illustrated in Figure 6.9. After the creation of



Class representative vector generator
Representative Class Vectors
Labeled Data
Membership by distance model
Suggested Words
Unlabeled Data
Labeled Data
Membership by dissimilar exclusion model
Suggested Words
Unlabeled Data
Ontology & Labeled Instances
Law Cases
Preprocessing
Labeled Data
Set expansion based model
Suggested Words
Cosine Similarity based Sorting
Performance Calculator
Ensemble Model
Labeled Data
Semi-supervised K-Means clustering based model
Suggested Words
Unlabeled Data
Labeled Data
Semi-supervised hierarchical clustering based model
Suggested Words
Unlabeled Data

Figure 6.8: Flow of Semi-Supervised Instance Population of an Ontology using Word Vector Embeddings

sub-ontology, we manually populated the ontology with seed instances with collaboration of domain experts and knowledge engineers.

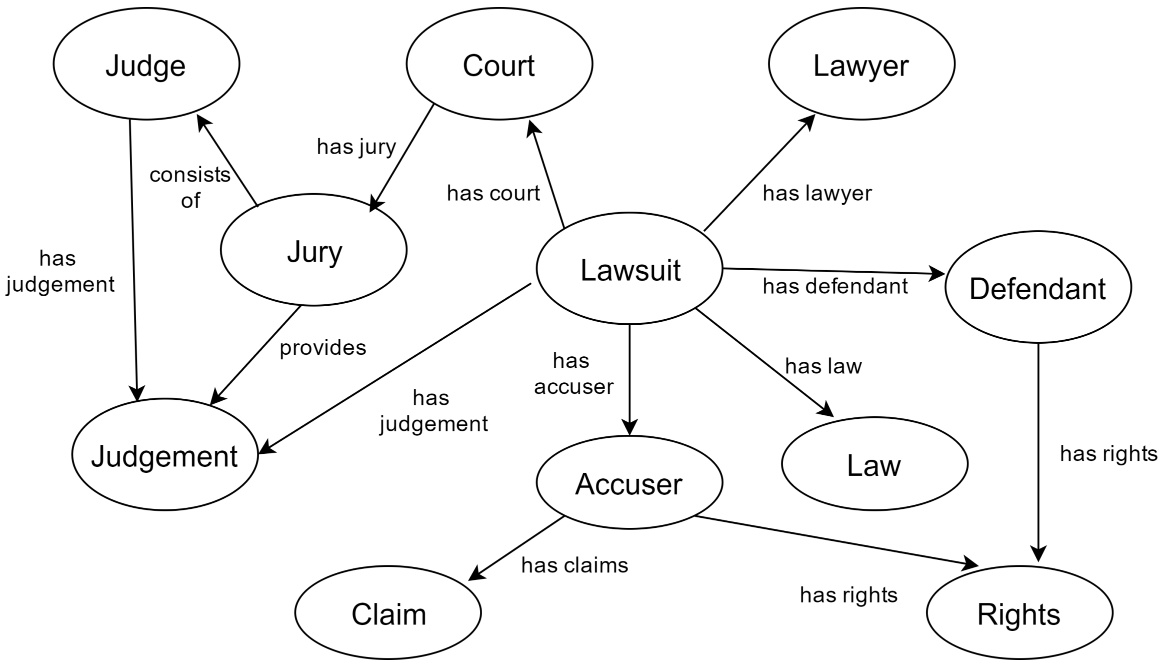


Figure 6.9: Ontology Sub-section used for the Population

6.6.2. Training word Embeddings

The word embeddings method used in this study was built using a *word2vec* model. We obtained a large legal text corpus from Findlaw [165] and built a *word2vec* model using the corpus. The reason for selecting word2vec word embedding for this study is the success demonstrated by other studies such as [13] and [101] in the legal domain that uses word2vec as the word embedding method. The text corpus consisted of legal cases under 78 law categories. In creating the legal text corpus, we used the *Stanford CoreNLP* for preprocessing the text with tokenizing and sentence splitting. Following are the important parameters we specified in training the model.

- size (dimensionality): 200
- context window size: 10

- learning model: CBOW
- min-count: 5
- training algorithm: hierarchical softmax

6.6.3. Deriving Representative Class Vectors

Ontology classes are sets of homogeneous instance objects that can be converted to a vector space by word vector embeddings. A methodology to derive a representative vector for ontology classes, whose instances were mapped to a vector space is presented in [13]. We followed the same approach and started by deriving five candidate vectors which are then used to train a machine learning model that would calculate a representative vector for each of the classes in the selected sub-ontology shown in Figure 6.9. In the following sections, we describe in-depth, the manner in how we used this derived class vectors in our proposed methodology.

6.6.4. Instance Corpus for Ontology Population

In order to perform semi-supervised ontology population, we used legal cases from Findlaw [165] to create an instance corpus. We performed *Stanford CoreNLP* based preprocessing on the raw text with tokenizing and sentence splitting to generate the instance corpus. This legal corpus was used in the subsequent models for the purpose of ontology population.

6.6.5. Candidate Model Building

Based on the aforementioned components, we built five candidate models for semi-supervised population of the ontology. The five models are illustrated below:

- Membership by distance model (M_1)
- Membership by dissimilar exclusion model (M_2)
- Set expansion based model (M_3)
- Semi-supervised K-Means clustering based model (M_4)
- Semi-supervised hierarchical clustering based model (M_5)

A. Membership by Distance Model (M_1)

In this model, the candidate vectors for the ontology were generated from the instance corpus based on the minimum distance to the representative class vector derived in Section 6.6.3. In taking the vector similarity, we used cosine similarity. Given an instance i which has the vector embedding X_i, Equation 6.15 describes which class the particular instance belongs to.

$$C_{M1} = \left\{ j \;\middle|\; \underset{c_j \in C}{\operatorname{argmax}} \left\{ \frac{X_i . c_j}{|X_i||c_j|} \right\} \right\} \tag{6.15}$$

Here, the set C denotes the set of representative class vectors. C_{M1} is the selected class index of the instance i out of class set C.

B. Membership by dissimilar exclusion model (M_2)

In this model, we used the word2vec based dissimilar exclusion method in identifying the membership of a particular instance to a given class. This is a utilization of an internal method of word2vec where given a set of members, it would return the member that should be removed from the set in order to increase the set cohesion. For example, given the set of instances: *breakfast*, *cereal*, *dinner* and *lunch*, the word2vec dissimilar exclusion method would identify the instance *cereal* as the item that should be removed from the set to increase the set cohesion. We define this method as shown in Equation 6.16 where S is the set provided and e is the member selected to be excluded.

$$e = Exclusion(S) \tag{6.16}$$

We used the Equation 6.17 to decide whether the instance i should belong to class j. Here, S_j is the seed set of class j and X_i is the vector representation of the instance i. If the value $E_{i,j}$ gets evaluated to $TRUE$ we declare that instance i should belong to class j under model M_2.

$$E_{i,j} = \left\{ e \in S_j \middle| e = Exclusion(S_j \cap X_i) \right\} \tag{6.17}$$

When using the aforementioned method in identifying the membership of an instance, there is a possibility of getting more than one class for a given instance as a possible parent class. Hence;

$$C_{M2} = \left\{ C_k \,\middle|\, 0 < k \leqslant N \right\} \tag{6.18}$$

Here in Equation 6.18, C_{M2} is the set of classes for a given instance i. C_k denote the common representation of those set of classes and N denotes the total number of classes we have in the ontology.

C. Set Expansion Based Model (M_3)

For the purpose of set expansion based model, we selected the algorithm presented in [20], which was built on the earlier algorithm described in [10]. The rationale behind this selection is the fact that as per [20], WordNet [21] based linguistic processes are reliable due to the fact that the WordNet lexicon was built on the knowledge of expert linguists.

In this model, the idea is to increase the ontology class instances based on a WordNet hierarchy based expansion. Simply put, it discovers the WordNet $synsets$ pertaining to the seed words and proceeds up the hierarchy to find the minimum common ancestors for each of the senses of the words. Next, the most common word sense is selected by majority. The relevant rooted tree is extracted and the gazetteer list of that rooted synset tree is created. The gazetteer list is subjected to a set subtraction of the original seed set. The set intersection of the remaining set with the candidate word set is declared to be the word set assigned to the given class. However, it should be noted that as we showed in model M_2, after running the set expansion algorithm, one candidate instance may be tentatively assigned to more than one class.

D. Semi-Supervised K-Means Clustering Based Model (M_4)

Out of the models proposed in this study so far, this model is the first semi-supervised model. First, the seed instances are put together with the unlabeled data from instance corpus. Let $N_{labeled}$ be the number of labeled (seed) instances and $N_{unlabeled}$ be the total number of unlabeled instances. Thus, by mixing up the labeled and unlabeled data, we get a total of $N_{labeled} + N_{unlabeled}$ number of instances. Next all the instances are subjected to the k-means algorithm where k is selected to the the same, as the number of classes in the ontology.

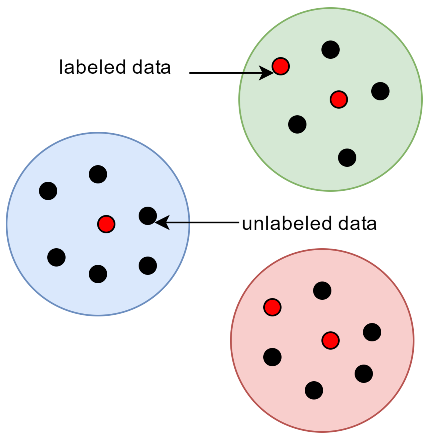


Figure 6.10: An illustration of k-means clustering with k = 3.

Once the k-means clustering is finished, primary class cluster assignment for cluster L is done by voting of seed instances according to Equation 6.19, where C is the set of ontology classes, c_j is the jth class from C, y_i is the ith instance from L, and d_i is defined according to Equation 6.20.

$$C_l = \left\{ j \,\middle|\, \underset{c_j \in C}{\operatorname{argmax}} \left\{ \sum_{y_i \in L} d_i \right\} \right\} \tag{6.19}$$

$$d_i = \begin{cases} 1 & \text{if } y_i \in c_j \\ 0 & \text{otherwise} \end{cases} \tag{6.20}$$

At this point, it should be noted that there can be three situations where it is possible to not get a c_l value assigned to some class L by Equation 6.19 without ambiguity: (1) L not having any seed instances to vote. (2) L has multiple seed instances but the majority voting ended in a tie. (3) Two (or more) clusters, claim the same class. To solve these problems we defined Equation 6.21, which selected the unassigned class that is closest to an unassigned cluster. Here, an unassigned cluster L' is considered. C' is the set of representative class vectors of unassigned classes. $C_{l'}$ is the selected class index of the cluster L'.

$$C_{l'} = \left\{ j \,\middle|\, \underset{c_j \in C'}{\operatorname{argmax}} \left\{ \sum_{x_i \in L'} \left\{ \frac{X_i.c_j}{|X_i||c_j|} \right\} \right\} \right\} \tag{6.21}$$

The first problem to be solved is the problem of L having multiple seed instances but the majority voting ending in a tie. In this case the C' of Equation 6.21 is limited to the set intersection of tied classes and unassigned classes. Next, the problem of Two (or more) clusters, claiming the same class is solved. In this case C' of Equation 6.21 is limited to the contested class. These steps are repeated until there is an iteration where there are no new assignments. Finally, all the remaining unassigned classes are put in c' and Equation 6.21 is executed repetitively with tie breaking, done with precedence until all the clusters are uniquely assigned to some class.

E. Semi-Supervised Hierarchical Clustering Based Model (M_5)

The next model being used is a semi-supervised method based on hierarchical clustering. Hierarchical clustering is a method of cluster analysis which seeks to build a hierarchy of clusters. We built a model which creates such hierarchy of clusters using the word embeddings taken from the word2vec model,of the entire corpus similar to the process in Section 6.6.5.D. In this model, we extracted the slice of hierarchical clusters such that the number of clusters in the slice is equal to the number of classes in the sub-ontology. Next, the cluster-class assignment was done similar to the process in 6.6.5.D.

Given an unlabeled instance Y, let $M_{ensemble}$ be a $p \times n$ matrix where n denotes the number of classes in the ontology and p denotes the number of basic models. Each column of the matrix corresponds to a class in the ontology and each row corresponds to a model, while each $m_{i,j}$ is derived from Equation 6.23.

$$M_{ensemble} = \begin{bmatrix} m_{1,1} & m_{1,2} & \dots & m_{1,n} \\ m_{2,1} & m_{2,2} & \dots & m_{2,n} \\ m_{3,1} & m_{3,2} & \dots & m_{3,n} \\ \dots & \dots & \dots & \dots \\ m_{p,1} & m_{p,2} & \dots & m_{p,n} \end{bmatrix} \tag{6.22}$$

$$m_{i,j} = \begin{cases} 1 & \text{if } Y \in M_i \\ 0 & \text{otherwise} \end{cases} \tag{6.23}$$

Let$M_{weights}$ be the p length vector which defines the weights of each model calculated by equation 6.31.

$$M_{weights} = \begin{bmatrix} w_1 & w_2 & w_3 & \dots & w_p \end{bmatrix} \tag{6.24}$$

Then we calculate the total score vector for the instance Y by,

$$S = M_{weights}.M_{ensemble} \tag{6.25}$$

Here, S is the score vector of size n, where element i in the vector denotes the total score for instance Y for the membership in Class C_i. Next, we selected the class with the highest membership score as the parent class of instance Y. It is illustrated in Equation 6.26.

$$C_{M_{ensemble}} = \left\{ i \,\middle|\, \underset{S_{C_i} \in S}{\operatorname{argmax}} \left\{ S_{C_i} \right\} \right\} \tag{6.26}$$

With that, we got the final class of the instance Y. Hence, we populate that selected class with the instance Y.

6.7. Information Retrieval Module

The main idea here is to use semantic similarity measures and create a document vector for each of the documents and generate a vector space. This gives the general idea of creating a vector space for documents, that will help us retrieve the documents with a certain similarity measure, that is based on Euclidean Distance between each of the document vectors.

This section describes the research that was carried out for this study. Each section below, addresses a component in the overall methodology. Initially, we collected over 2500 legal case documents for this study from the FindLaw [36] website using multi-threaded webcrawlers. Hereafter in this study, we refer to this legal case document collection as the text corpus. An overview of the methodology we propose in this system is illustrated in Figure6.11.

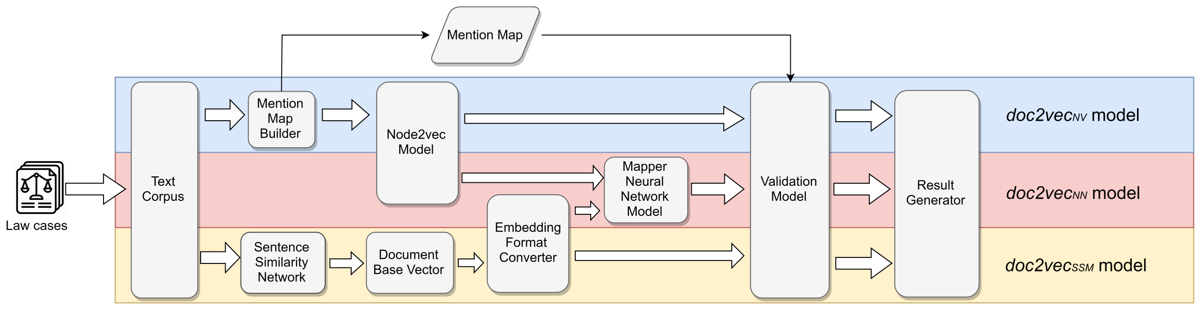


Figure 6.11: Flow diagram for the Overall Methodology

The following sub sections describe the steps that were carried out in this study, where we generate three unique document vector models: one is generated with raw document references, another using a neural network, and the other one is generated using sentence ranking derived from the domain specific semantic similarity measures, presented in [101]. Each document vector model is used to train a neural network with n-fold cross validation, which is then tested against a golden standard benchmark to generate the final results.

6.7.1. Document Relevance Mention Map

For this study, we defined a single legal case as a document unit, which gives us a granularity level [166] to contain many legal case documents pertaining to different legal areas. Given that, legal cases are reports of the proceedings of a lawsuit. It is possible to observe that prior legal cases are mentioned in them as case law references. By definition, these are the legal cases that the lawyers involved in the case, deemed relevant to the case at hand. Given that this study is also concerned with finding relevant legal cases for a selected legal case, it is imperative that we use the mentioned legal cases for the training of the models. Hence, we defined the *inverted index* structure, where for each document, we maintained a list of documents, which were referenced in that text. A list of this nature is called the *postings list*, where each element in the list is defined as a *posting* [35]. We use a *Dictionary* data structure to store the document vocabulary and a linked list for each of the posting lists. This overall structure is illustrated in Figure 6.12. The size of the document vocabulary (n), is the number of document units contained in the text corpus.

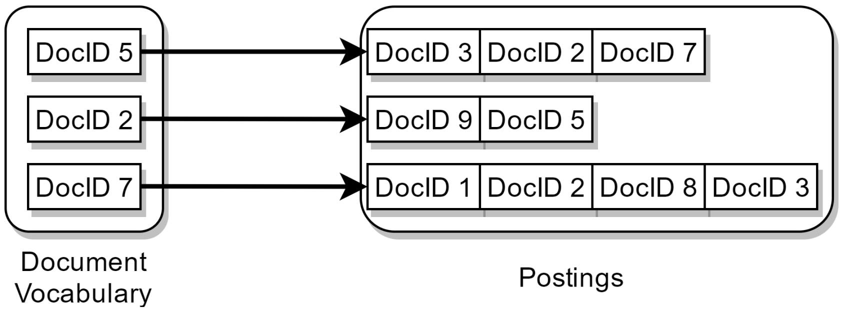


Figure 6.12: Document Relevance Mention Map

The listing of mentions and references of other legal cases was a non-trivial Information Extraction (IE) task as discussed by [35]. This was mainly due to the difference in abbreviations and legal jargon that were used in online repositories. The first problem was the lack of a standard and proper naming convention for the legal case names. For example, depending on the case, the word *corporation* was shortened as *corp* or *co.* even when referring to the same legal case title, despite what is on record as the proper legal case name. This problem was solved by regular expression-based Information Extraction [32]. The second problem was the legal name reference using abbreviations of the case name in the text body. For example *Office of Workers' Compensation Programs* was referred as *OWCP*, in the body text of the cases that referred cases pertaining that entity despite the fact that all the cases that were being referred were using the full name of the entity in their titles. To solve this problem, extensive searching on the Findlaw Search Engine was done, and mention names were matched with the legal case document names.

This mapping from each document, to other documents which refer to the original document, is called the *mention map* (M). Equation 6.27 shows the formal definition of M, which contains n number of keys. Each key m, has an associated postings list l, that contains the document IDs which have referenced in m.

$$M = \begin{bmatrix} m_1 & m_2 & m_3 & \dots & m_n \\ l_1 & l_2 & l_3 & \dots & l_n \end{bmatrix} \tag{6.27}$$

This is the base component of our study, which will be directly used as input to train the $doc2vec_{NV}$ model and to validate the neural network as shown in Section 6.7.9. For the $doc2vec_{SSM}$ model, we carried out a number of advanced computational steps which will be discussed in the following sections. Incorporating these two models into another unique model, using a neural network, is given in the $doc2vec_{NN}$ model. In Section **??**, we compare and contrast these three models and show the improvement of accuracy that can be gained by the following domain specific enhancements.

6.7.2. Sentence Similarity Graph Network

In a text document, sentences play a vital role to its semantic structure [167]. This is mainly due to the fact that the semantic information contained in the words in sentences are an accurate representation of the overall semantics of the document. Thus, in the attempts to represent a document as a feature vector, one of the most common methods is the *bag of words* [53] approach, which yields acceptable results even-though each word is considered independent of the surrounding words in the context. In this study, we created a sentence similarity graph network using the semantic similarity measures between sentences by the *TextRank* algorithm [152] described in Section 4.7. We used a threshold of 0.5 to determine the level of similarity between nodes in the graph.

We created a document corpus, which is a subset of the entire text corpus, where we picked the most important sentences in each of the documents, from the sentence similarity graph network. We selected the k most important sentences within a document, using the sentence similarity graph network generated, with $|S| \geq k$, where S is the set of sentences within a document. Hereinafter, this subset will be referred to as the *document corpus*.

6.7.3. Text Preprocessing

First, we pre-processed the *document corpus* to clean the text of unwanted characters and common words, in order to obtain the optimal size for the final vocabulary V. The pipeline that we used in this study is illustrated in Figure6.13. In the linguistic preprocessing step, we used lemmatizing and case-folding to lowercase, as our primary Natural Language Processing(NLP) techniques. We used the Stanford Core NLP [103] library for this purpose.

6.7.4. Model Accuracy Measure

After building the aforementioned models, we evaluated the accuracy of each model. As each model outputs an unordered set of suggested words, we sorted them using the Neural Network, trained according to the methodology proposed in [101]. Upon completing the sorting, we applied a threshold to select the best candidates. Finally, we measured each model's accuracy as below. For this task, we involved with domain experts and knowledge engineers. For a given model M_i in the context of class j:

$$Precision_{M_{i,j}} = \frac{W_{M_{i,j}} \cap W_{j,g}}{W_{M_{i,j}}} \tag{6.28}$$

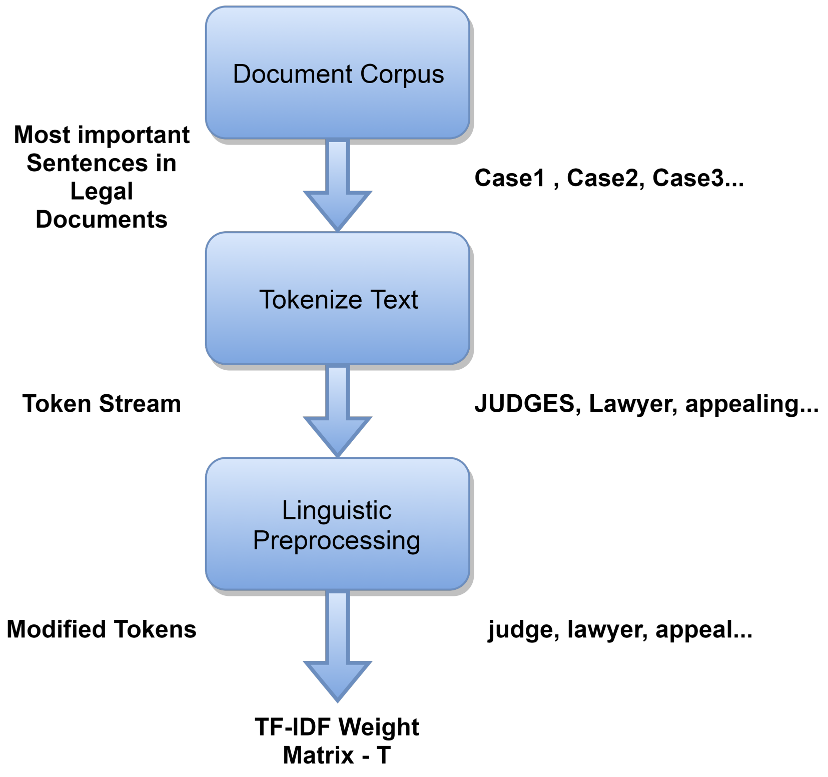


Figure 6.13: Preprocessing Pipeline

$$Recall_{M_{i,j}} = \frac{W_{M_{i,j}} \cap W_{j,g}}{W_{j,g}} \tag{6.29}$$

Here, W_{M_i} denotes words by the model M_i and $W_{j,g}$ denotes the set of the words proposed by domain experts that needs to be the golden standard for class j. The model precision and model recall of M_i was calculated by averaging the class values for precision and recall of those models.

$$F1_{M_i} = 2.\frac{Precision_{M_i}.Recall_{M_i}}{Precision_{M_i} + Recall_{M_i}} \tag{6.30}$$

6.7.5. Ensemble Model

Next, we came up with an ensemble model based on the models identified earlier. In the task of creating the ensemble model, we allocated a candidate weight for each model based on each model's $F1$ measure as calculated in the previous step.

Let M_i be a model, out of the obtained models, and let $F1_{M_i}$ be the F1 measure of model M_i. Hence with the models in consideration, weight of the model W_i is calculated as shown in equation 6.31, where p is the total number of models.

$$W_i = \frac{F1_i}{\sum_{i=1}^{p} F1_i} \tag{6.31}$$

As identified above, upon calculating the weight of each model, we created the ensemble model as shown in equation 6.22.

6.7.6. Document Base Vector Creation

We ran TF-IDF [59] on the *document corpus* and built a TF-IDF weight matrix T between the terms and documents. T is a $v \times n$ matrix, where v is the size of vocabulary vector V. Matrix T is shown in Equation 6.32, where $t_{i,j}$ is the TF– IDF value of term i in document j of corpus D calculated according to Equation 5.3.

$$T = \begin{bmatrix} t_{1,1} & t_{1,2} & t_{1,3} & \dots & t_{1,n} \\ t_{2,1} & t_{2,2} & t_{2,3} & \dots & t_{2,n} \\ \dots & \dots & \dots & \dots & \dots \\ t_{v,1} & t_{v,2} & t_{v,3} & \dots & t_{v,n} \end{bmatrix} \tag{6.32}$$

Next we defined GTF (Global Term Frequency) as shown in Equation 6.33. GTF is an extension of Equation 5.1 that was used to calculate TF. Here, n is the number of legal case documents considered.

$$GTF_{t,D} = \frac{\sum_{i=1}^{n} TF_{t,i}}{n} \tag{6.33}$$

Then we scaled both the GTF and IDF values as according to Scalings 6.34 and 6.35.

$$0 \leqslant GTF \leqslant 1 \tag{6.34}$$

$$0 \leqslant IDF \leqslant 1 \tag{6.35}$$

Next we pruned the GTF and IDF values as given in Equations 6.37 and 6.39, where α values are given in Equation 6.40. After scaling the GTF and IDF values obtained from above, we calculate the mean separately for both of them, as μ_{GTF} and μ_{IDF}. Next step is to calculate the standard deviation separately for both of them, as σ_{GTF} and σ_{IDF}. The α value represents the factor by which the standard deviation is multiplied, and the range of the dataset is selected. This is depicted in Figure6.14.

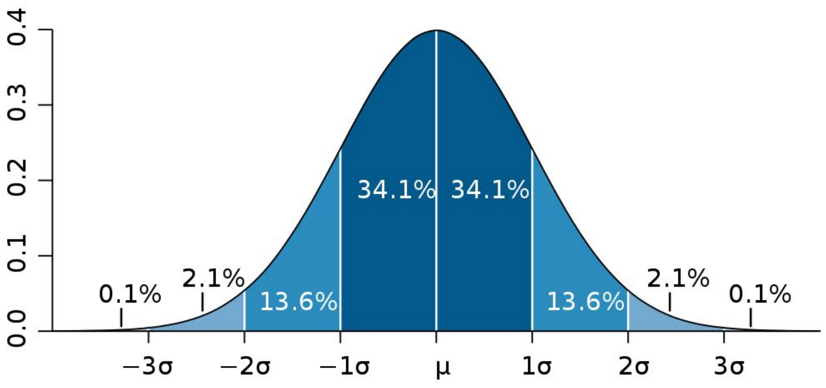


Figure 6.14: Distribution of GTF and IDF values with a factor of standard deviation

$$r_{GTF} = \alpha \times \sigma_{GTF} \tag{6.36}$$

$$GTF_{t,D} = \begin{cases} x_{t,D} & \text{if } \mu_{GTF} - r_{GTF} \leqslant x_{t,D} \leqslant \mu_{GTF} + r_{GTF} \\ 0, & \text{otherwise} \end{cases} \tag{6.37}$$

$$r_{IDF} = \alpha \times \sigma_{IDF} \tag{6.38}$$

$$IDF_{t,D} = \begin{cases} x_{t,D} & \text{if } \mu_{IDF} - r_{IDF} \leqslant x_{t,D} \leqslant \mu_{IDF} + r_{IDF} \\ 0, & \text{otherwise} \end{cases} \tag{6.39}$$

$$\alpha = [0.1, 0.5, 1, 2, 3] \tag{6.40}$$

Using GTF, we defined GTF-IDF in Equation 6.41 as an extension to TD-IDF given in Equation 5.3, to generate a $GTF\text{–}IDF_{t,D}$ value for each term in the vocabulary V. In GTF-IDF, the IDF algorithm is the same as Equation 5.2. The objective of GTF-IDF is to identify the words that are most significant to the domain D.

$$GTF\text{–}IDF_{t,D} = GTF_{t,D} \times IDF_{t,D} \tag{6.41}$$

We defined V' by sorting V by the descending order of $GTF\text{–}IDF_{t,D}$ values and obtained the common base vector template (B) as shown in Equation 6.42. B is a word vector of length p, such that the p number of words that are most significant to the domain D are contained in B. Thus, B was obtained by taking the first p elements of the sorted V'.

$$B = \{term_1, term_2, ..., term_p\} \tag{6.42}$$

6.7.7. Sentence Similarity Based Document Vector Representation

Next, we created a vector representation for documents in the text corpus. As mentioned in Section 6.7.6, the vector representation of a document was a p dimensional vector, representing the most important p terms from the context on the entire text corpus, as shown in Equation 6.42. As mentioned in Section 6.7.2, we selected the k most important sentences within a document, with $|S| \geq k$, where S is the set of sentences within a document. For the jth document, we defined R_j to be the total set of unique words in the selected k sentences. We defined the $Seek$ function as shown in Equation 6.43, which would return the index i of word w given a vocabulary of words U.

$$i = Seek(w, U) \tag{6.43}$$

Finally, we defined the document vector representation H_j for the jth document as shown in Equation 6.44, by calculating each $h_{j,i}$ as given in Equation 6.45, where b_i is the ith element of the document vector template B created by Equation 6.42, and $t_{a,j}$ is the element at row a column j of matrix T defined in Equation 6.32. Each document vector H_j was normalized using the L2 Normalization [168].

$$H_j = \{h_{j,1}, h_{j,2}, ..., h_{j,i}, ..., h_{j,p}\} \tag{6.44}$$

$$h_{j,i} = \begin{cases} t_{a,j} & \text{if } b_i \in R_j, a = Seek(b_i, V) \\ 0, & \text{otherwise} \end{cases} \tag{6.45}$$

6.7.8. Scalable Feature Learning Node2vec Model

As depicted in Figure6.11, this study built two unique models: $doc2vec_{NV}$ model and the $doc2vec_{SSM}$ model. The $doc2vec_{NV}$model was generated using an algorithm known as Node2vec [169], which is a wrap around the word2vec model introduced by Thomas Mikolov [9]. The node2vec algorithm is a scalable feature learning technique for networks, which learns continuous representations for nodes in any (un)directed or (un)weighted graph.

The input for this algorithm was an Edgelist, which contained the list of edges generated from the mention map in Section 6.12. Each document in the document vocabulary was paired with each of their references separately, and that was used to generate the list of edges. If *DocID 2* had referenced *DocID 9* and *DocID 5*, then the Edgelist would be pairs as *DocID 2, DocID 9*, and *DocID 2, DocID 5*

Finally, the output was a vector space that contains a set of feature vectors for the list of documents in the legal text corpus.

6.7.9. Mapper Neural Network Model

The Mapper Neural Network model $doc2vec_{NN}$, was trained using the both models: $doc2vec_{NV}$ model and the $doc2vec_{SSM}$ model. These two models were trained separately on different vector spaces, but with the same set of document IDs. Therefore, it is mandatory to build a model that could incorporate the different features in both of these vector spaces and produce the document ID of a given legal document, when the corresponding document ID in the other vector field, is provided. This process is depicted in Figure6.15

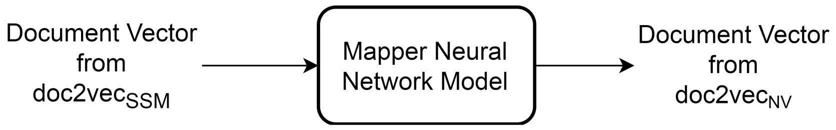

Document Vector from doc2vec$_{SSM}$
Mapper Neural Network Model
Document Vector from doc2vec$_{NV}$

Figure 6.15: Mapper Neural Network Input and Output

6.8. Linguistic Rules and Gazetteer Lists

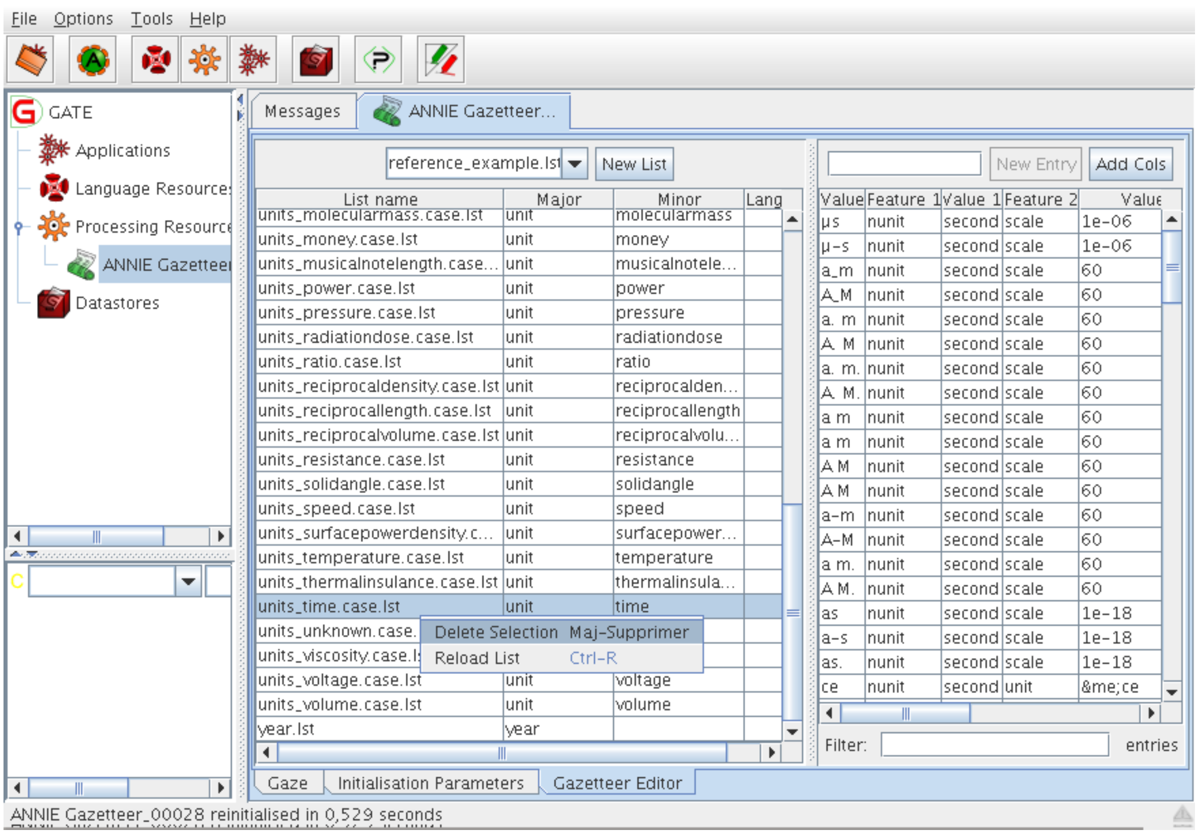
When a legal case document is given, we should be able to extract the necessary information off the text. This process is called information extraction, where certain concepts or classes will be looked for in the given piece of text, and the instances of each of those concepts or classes will be extracted from the text.

We have used JAPE rules for this purpose, where it will identify certain structures within the text to extract certain types of information as necessary. The GATE [5] Architecture is being used to write JAPE rules and extract certain instances of classes.

This extraction process is guided by the Legal Ontology generated, whereas instances of each of the concepts given in the ontology will be extracted from the text document, whilst maintaining the relationships and axioms, as given in the ontology.

Once these instances are being extracted, it will be stored in a knowledge base, so that it can be used further in the process towards a query answering system for the legal domain.

Also, we used these instances, and the extraction process to generate gazetteer lists, which can be used within this project and outside of this project as well. Figure Figure6.16 shows an example use of the GATE Gazetteer List component to edit the Annie Gazetteer list.



List name	Major	Minor	Lang
units_molecularmass.case.lst	unit	molecularmass	
units_money.case.lst	unit	money	
units_musicalnotelength.case...	unit	musicalnotele...	
units_power.case.lst	unit	power	
units_pressure.case.lst	unit	pressure	
units_radiationdose.case.lst	unit	radiationdose	
units_ratio.case.lst	unit	ratio	
units_reciprocaldensity.case.lst	unit	reciprocalden...	
units_reciprocallength.case.lst	unit	reciprocallength	
units_reciprocalvolume.case.lst	unit	reciprocalvolu...	
units_resistance.case.lst	unit	resistance	
units_solidangle.case.lst	unit	solidangle	
units_speed.case.lst	unit	speed	
units_surfacepowerdensity.c...	unit	surfacepower...	
units_temperature.case.lst	unit	temperature	
units_thermalinsulance.case.lst	unit	thermalinsula...	
units_time.case.lst	unit	time	
units_unknown.case.			
units_viscosity.case.l			
units_voltage.case.lst	unit	voltage	
units_volume.case.lst	unit	volume	
year.lst	year		

Value	Feature 1	Value 1	Feature 2	Value
µs	nunit	second	scale	1e-06
µ-s	nunit	second	scale	1e-06
a_m	nunit	second	scale	60
A_M	nunit	second	scale	60
a. m	nunit	second	scale	60
A. M	nunit	second	scale	60
a. m.	nunit	second	scale	60
A. M.	nunit	second	scale	60
a m	nunit	second	scale	60
a m	nunit	second	scale	60
A M	nunit	second	scale	60
A M	nunit	second	scale	60
a-m	nunit	second	scale	60
A-M	nunit	second	scale	60
a m.	nunit	second	scale	60
A M.	nunit	second	scale	60
as	nunit	second	scale	1e-18
a-s	nunit	second	scale	1e-18
as.	nunit	second	scale	1e-18
ce	nunit	second	unit	&me;ce

ANNIE Gazetteer_00028 reinitialised in 0,529 seconds

Figure 6.16: GATE - Annie Gazetteer Editor

[5] https://gate.ac.uk/

6.9. Information Extraction Module

This section gives the complete idea about the information extraction process being carried out in this system. The initial idea is to collect the legal document via web crawling, pre-process them and create a set of annotated document, to guide the information extraction process. This is used as a guidance for the extraction process alongside with the Ontology generated with the help of legal domain experts. Using the annotated legal documents, the necessary rules can be generated and stored whee these can be referred to as information extraction rules.

These rules are generated with the help of a tool named Protege, and the extraction rules can be generated with the help the annotated documents. This is then fed into the information extraction module generated in this system

In parallel, the legal documents and the legal knowledge is being used to generate a legal knowledge as explained in the this study. This is also guided by legal domain experts, and this is also fed into the information extraction module made in this system.

Finally, the information extraction module extracts the necessary information from the legal case texts and the extracted information is stored within a NOSQL Database. This is also used to create Gazetteer lists, which is being used later in the Query Processing System.

The information extraction module is also used to generate a feature vector for the documents. This is later used in the final query to categorize documents based on their document vectors.

6.10. Query Answering System

The query answering system is the final interface provided provided to general public or the paralegal, whichever user is using the system. This component supports the following main functionalities.

This section incorporates a vector space that is needed for the final query out generation process. This includes the document vector generated in Section 6.7 and the feature vector generated in Section 6.9. It combines these two vectors to generate the final document vector that is being used to build a vector space. With the help of this vector space, a neural network is being trained.

Next, the user will then enter a query to the query interface and then the system will take this method and query the trained neural network where this query is created as a vector and fed into the network. Finally, we try to get the closes vectors from the trained neural network, which includes a vector space of the legal document. The closest vectors to the query vector is returned by the system based on the Euclidean Distance. This will be the final output of the system.

7 Experimental Results

This section addresses the experiments and results being carried out in the development of the entire system. Each section is broken down into important aspects where the required information is gather, with experiment carried out and finally collecting all the required results.

7.1. Data Collection

The data collection of this system was carried out using web crawling as mentioned in the Methodology, where JSoup was used as a Java Library to support the collection process. The data was collected from the online repositories that contain United States Legal Information. We used the FindLaw website for this purpose where the necessary legal cases were extracted out of it.

Initially, we extracted 55000 legal case documents, which took over one week to finish extracting all the cases and their related contents. This was very heavy processing for the rest of the methodology, and due to this reason, we created some sample test cases that contained around 25000 legal case documents and 2500 legal case documents.

The case documents of 2500 was initially used as a proof of concept where later the same methodology was applied to the entire document corpus with higher and better performance machines for heavy processing.

7.2. Semantic Similarity

This section describes the results and experiments done in the semantic similarity process as explained in the methodology section.

7.2.1. Experiments

We got experts in the legal field to create a golden standard to test our models. The golden standard includes 100 concepts with each containing 5 words that are most related to the given concept in the legal domain picked from a pool of over 1500 words by the legal experts.

The accuracy levels of these experiments are measured in terms of precision and recall [1], which are common place measures in information retrieval. The logical functionality of these two are based on the comparison of an expected result and the effective result of the evaluated system.

If the Golden Standard Word Vector is G and the word vector returned by the model is W (Same naming conventions as Section 6.3.2.A), the recall in this study is calculated with equation 7.1. This measures the completeness of the model. Our recall calculation used the same function suggested in [1].

$$recall = \frac{|G \cap W|}{|G|} \tag{7.1}$$

The precision calculation in this study is not as clear cut as it is described in [1]. This is because in those systems the precision is only a matter of set membership and thus would simply be ratio between the correctly found similar words over the total number of returned words. However, in the case of word2vec models, it is the user who input the number of matching words to retrieve. Thus, it is wrong to use the total number of returned words to calculate the precision in cases where there is prefect recall. In the cases of imperfect recall, the classical precision is adequate.

While word2vec make precision calculation difficult as shown above, it also has a quality that makes finding the solution to that problem somewhat easy. That is the fact that the returning word list of similar words is sorted in the descending order. This is the same property that we used in the above Section 6.3.2.B to calculate the error. Thus it is logical to derive that the precision is given by equation 7.2 where err is the error calculated by equation 6.10.

$$precision = 1 - err \tag{7.2}$$

Table 7.1: Results comparison (P=Precision, R=Recall)

Model	k=20			k=50			k=100			k=200		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
$word2vec_G$	0.57	0.19	0.29	0.62	**0.33**	0.43	0.67	0.41	0.51	0.74	0.46	0.57
$word2vec_{LR}$	**0.75**	0.19	0.30	0.71	0.31	0.43	0.74	0.38	0.51	**0.77**	0.44	0.56
$word2vec_{LL}$	0.73	0.22	0.34	0.72	0.32	0.45	**0.75**	0.40	0.52	0.76	0.47	0.58
$word2vec_{LLS}$	0.66	**0.24**	**0.36**	**0.73**	**0.33**	**0.52**	0.72	**0.43**	**0.54**	0.74	**0.50**	**0.60**

7.2.2. Results

This section includes results obtained from the four different models ($word2vec_G$, $word2vec_{LR}$, $word2vec_{LL}$, and $word2vec_{LLS}$) as introduced in Section 6.3.2. Results shown in table 7.1 were obtained for different k values, where k is the number of words requested from each of the models. As expected, the F1 of each model increases with k, where the possibility of finding the correct similar words against the golden standard increases. Given that the task here is to return the expected set of words, the recall is more important than precision (i.e: False-Negatives are more harmful than False-Positives). In that light, it is obvious that the $word2vec_{LLS}$performs better than all other models because it consistently has the highest recall for all values of k. In addition to that, the $word2vec_{LLS}$model also has the highest $F1$ for all values of k, which is sufficient proof that the small loss in precision does not adversely affect the overall result.

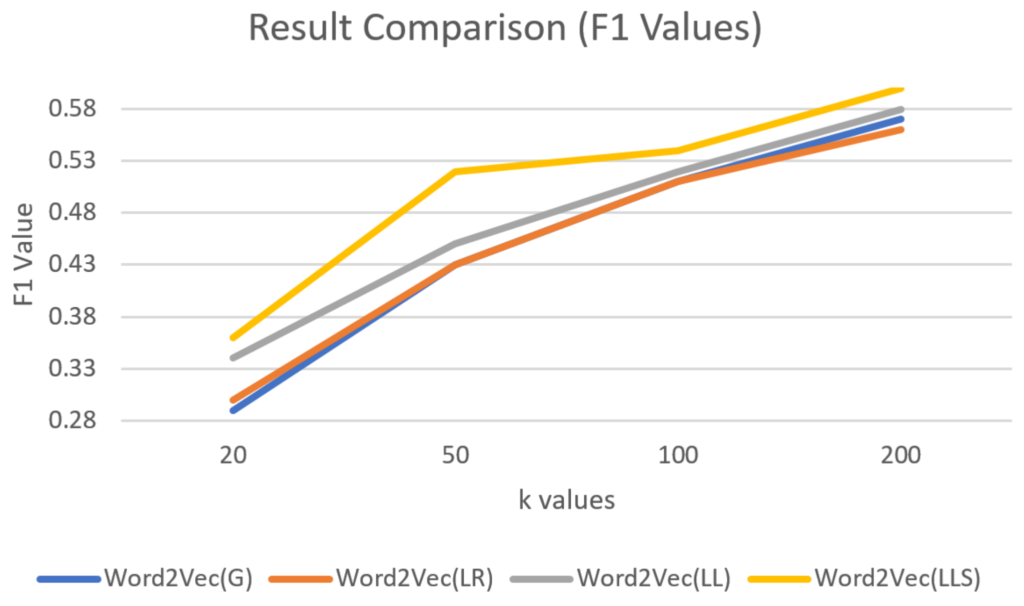


Figure 7.1: F1 value comparison

A graphical comparison of the changes in the $F1$ measure is shown in Figure 7.1. As shown, the domain specific models such as $word2vec_{LR}$, $word2vec_{LL}$, and $word2vec_{LLS}$, show better results than the general $word2vec_G$model. It should be noted that this performance enhancement has happened despite the fact that $word2vec_G$was trained on a text corpus 3 times bigger than the text corpora we have used in this study for the models $word2vec_{LR}$, $word2vec_{LL}$, and $word2vec_{LLS}$.

In the comparison of domain specific models, we can see a clear distinction between the $word2vec_{LR}$ and $word2vec_{LL}$ models, where the $word2vec_{LL}$ generally performs better. Further, it can be observed that the $word2vec_{LLS}$ model outperforms both $word2vec_{LR}$ and $word2vec_{LL}$ models.

7.3. Ontology Class Vector Generation

Results

We used a set of ontology classes seeded by legal experts and then expanded by the set expansion [20] algorithm under the guidance of the same legal experts. We report our findings below in the table 7.2, and in Figure7.2 we show a visual comparison of the same data. We illustrate the results that we obtained pertaining to ten prominent legal concept classes as well as the mean result across all the classes considered. We compare the representative vector proposed by us against the traditional representative vectors: average vector and median vector. All the results shown in the table are Euclidean distances obtained between the representative vector in question against the respective C_0 vector. It should be noted that lesser the distance between the representative vector and the respective C_0 vector, better the accuracy of the representative vector.

Table 7.2: Distance measures of the classes and the average over 10 classes

	Average Vector	Median Vector	Our Model
Appeal	5.37	8.32	**2.31**
Assault	4.73	11.53	**2.27**
Complaint	5.33	11.82	**2.46**
Defendant	4.60	9.51	**2.15**
Judge	3.52	4.56	**1.84**
Lawyer	3.89	6.33	**2.35**
Theft	4.56	8.49	**2.13**
Trademark	7.63	14.79	**2.81**
Victim	5.64	11.52	**2.54**
Violation	4.18	10.80	**2.09**
Witness	4.46	6.73	**2.34**
Class mean	1.31	1.51	**0.82**

It can be observed from the results that the traditional approach of taking the average vector as the representative vector of a class is outperforming the other traditional approach of using the median vector. However, our proposed method outperforms both the average and median vectors in all cases. For an example, considering the "Judge" class, it can be seen that our model vector perform 47.8% better than the average vector where it is 53.8% better in the "Complaint" class. As in an average case, it is evident from the results that our model has the lowest class mean distance compared with the traditional approaches where it is 37.4% and 45.7% in terms of the improvements with average and median vectors respectively.

7.4. Semi-Supervised Ontology Instance Population

Results

In testing our ensemble model, we used another instance corpus. In this corpus, we subdivided in the order of 70%, 20%, and 10% as the training set, the validation set, and the test set respectively. Training set was used in training the models individually. Validation set was used to fine tune the models. Finally, the test set was used in verifying the accuracy of the models. We report our findings below in the table 7.3, where we compare the individual models: membership by distance model(M_1), membership by dissimilar exclusion model(M_2), set expansion based model(M_3), k-means clustering based model(M_4), hierarchical clustering based model(M_5) and the ensemble model as a whole. In Fig. 7.3, we compare the precision, recall and F1 of each of the candidate models along with the ensemble model.

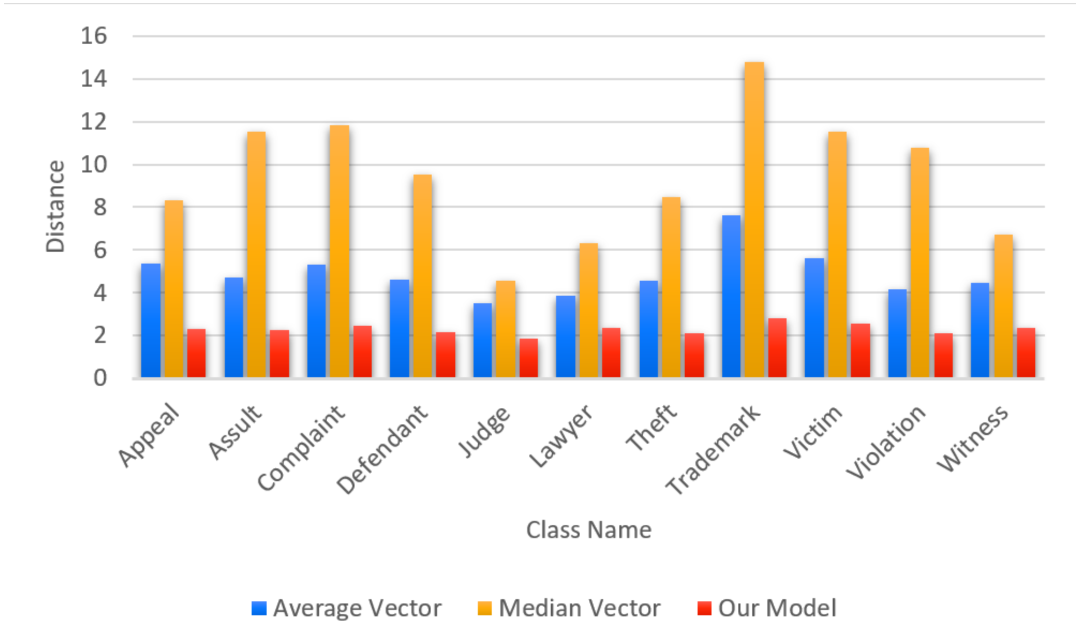


Figure 7.2: Comparison of the distance measures of representative vectors

Table 7.3: Comparison of performance of models

	Precision	Recall	F1
M_1	0.08	0.22	0.12
M_2	0.15	0.36	0.21
M_3	0.24	0.30	0.26
M_4	0.07	0.20	0.10
M_5	0.06	0.23	0.10
$M_{ensemble}$	**0.51**	**0.63**	**0.56**

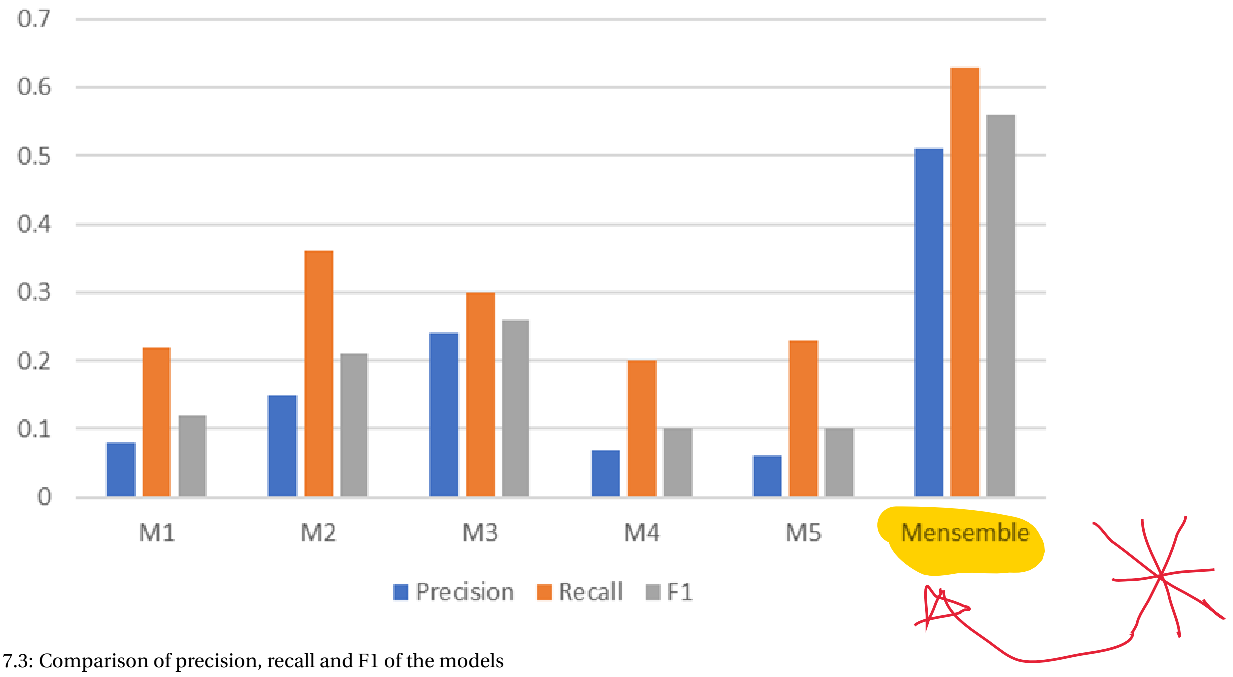


Figure 7.3: Comparison of precision, recall and F1 of the models

In defining the ensemble model, Equation 7.3 defines the calculated weights of each model in the order of models M1 to M5. These calculations are based on the above mentioned training set.

$$M_{weights} = \begin{bmatrix} 0.15 & 0.27 & 0.33 & 0.13 & 0.12 \end{bmatrix} \tag{7.3}$$

As can be seen from Table 7.3, our ensemble model's F1 has been improved by 0.30, compared to the best of the candidate models. Hence, from the results obtained, as a proof of concept, we can demonstrate that word embeddings can be used effectively in semi-supervised ontology population.

7.5. Information Retrieval Module

7.5.1. Experiments

As mentioned in Section 6.7.9, we generated the $doc2vec_{NV}$ model and the $doc2vec_{SSM}$ model separately, which were later used to obtained the $doc2vec_{NN}$ model. In this study, our experiments are to compare and contrast the accuracy levels of these three models to varying p values, where p is the dimension of the document base vector B, as given in Equation 6.42. The p values used in this study were; 250, 500, 750, 1000 and 2000. The accuracy measures we use for this study are based on recall [1], of which the formula is given by Equation 7.4, where $\{Relevant\}$ is the set of relevant documents taken from the golden standard and $\{Retrieved\}$ is the set of documents requested from the model. $\{Relevant\}$ is the number of unique references each *DocID* has got in the document mention map, which is depicted in Figure6.12 in section 6.12. In other words, this is the length of l (posting list) of each document, according to Equation 6.27.

$$Recall = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Relevant\}|} \tag{7.4}$$

For each of the different models, we used ten-fold cross validation [170] to measure the accuracy levels. The final results were further validated with the help of legal domain experts, to ensure that the results were accurate as expected.

7.5.2. Results

The results obtained in this study is given in Table 7.4, with varying α and p values. This section of the study contains results obtained from the three models, where the recall values are calculated based on the golden standard measure, as mentioned in Section 7.5.1.

Table 7.4: Results comparison ($doc2vec_{SSM}$, $doc2vec_{NV}$, $doc2vec_{NN}$)

α Value		0.1			0.5			1			2			3		
Model		SSM	NV	NN	SSM	NV	NN	SSM	NV	NN	SSM	NV	NN	SSM	NV	NN
p Value	p=250	16.26	**87.70**	30.05	33.89	**87.70**	69.70	35.01	**87.70**	84.04	36.04	**87.70**	81.13	39.21	87.70	**90.34**
	p=500	23.27	**87.70**	41.80	39.17	**87.70**	85.61	39.80	87.70	**88.03**	39.60	87.70	**89.96**	42.89	87.70	**90.00**
	p=750	25.96	**87.70**	47.85	41.22	87.70	**90.58**	44.30	87.70	**92.33**	42.05	87.70	**92.68**	43.84	87.70	**93.16**
	p=1000	26.86	**87.70**	49.14	42.65	**87.70**	83.91	46.71	**87.70**	87.32	44.95	87.70	**89.17**	44.14	87.70	**91.64**
	p=2000	28.03	**87.70**	54.77	43.79	87.70	**90.44**	49.33	87.70	**90.15**	52.36	87.70	**93.81**	51.59	87.70	**88.32**

8 Conclusion and Future Work

8.1. Future Directions of Information Extraction

The legal domain in the digital world is an emerging topic at present, where text mining techniques like information extraction show greater potential towards major applications in the legal domain and elsewhere. Legal Information Extraction systems is expected to grow in different directions, leading to many major directions. In this section, we highlight some major areas where legal information extraction systems can contribute to, in order to emerge forward in this field of study.

8.1.1. A semantic web for legal domain

According to the World Wide Consortium (W3C), a "the Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries" [171]. The idea is to provide a framework that incorporates semantics and domain specificity, to develop a web that is more robust and efficient. The semantic web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries [172]. Tim Berner's Lee who came up with the title *Semantic Web* mentioned that it is a web of data that can be processed by a machine.

The difficulties faced, are only a subset of what is actually faced when trying to map the legal domain, to extract or process necessary legal information in various contexts. This is mainly because legal concepts are not discrete, but make up a dynamic continuum between common sense terms, specific technical use, and professional knowledge, in an evolving institutional reality [173]. This study also attempts to addresses the effort given by researchers to build a semantic web for the legal domain and the future directions and progress of the existing systems. Legal information extraction system can help develop semantically better legal systems that will indeed help developing the semantic web.

8.1.2. Legal Ontologies

An ontology is a "formal and explicit specification of a shared conceptualization [174]." Almost all legal information extraction processes are guided by domain expert annotated models, as defined in Section 4.2.2, where ontologies are an example for it. The accuracy and quality of these annotated models or ontologies define the accuracy and quality of the information extraction process. Ontologies are the main mechanism for domain specific knowledge representation in the Semantic Web context, but their manual maintenance is expensive and error-prone [175]. With the help of legal information extraction systems, the quality of ontologies can be evaluated, as well as the development of ontologies will be much more efficient in future.

8.1.3. Legal Ontology Population

Populating ontologies have been a very cumbersome task due to the difficulty in properly extracting relevant pieces of information related to the ontology model. Many researchers have approached this task in manual and automatic approaches where there success rates still need further improvements. Jayawardena et al. [14] has introduced a novel way in populating ontologies using semi-supervised approaches, where Bruckschen et al. [175] has proposed a method to automatically update legal ontologies from legal texts through the Named

Entity Recognition task. Ontology population requires identification and extraction of entities from a given piece of text and legal information extraction systems will be widely used for this purpose, for the legal domain.

8.1.4. Case Based Reasoning

In general, Case Based Reasoning (CBR) is a broad concept where problems are being solved based on solutions pf similar past problems. In the legal domain, legal cases are solved via past legal cases which provides the guidance and direction towards solving a legal case. For this reason, legal officers require the knowledge relating to pasts cases to make proper judgments and interpretations [176]. Information extraction and retrieval plays a major role in CBR tasks, where Sugathadasa et al. has used a neural network absed approach to retrieve similar and relevant legal cases when a legal case document is being queried. Legal information extraction systems will indeed provide better directions for CBR activities in the legal domain.

8.2. Synergistic Union of Word2vec and Lexicon for Domain Specific Semantic Similarity

Conclusion and Future Works

The hypothesis of this study had three main claims and each of these claims were justified by the results presented. The first claim of the hypothesis is that a word embedding model trained on a small domain specific corpus can out perform a word embedding model trained on a large but generic corpus. The success of $word2vec_{LR}$model over the $word2vec_{G}$model justifies this claim. In Section **??** we proposed the second claim: word lemmatization, which removes inflected forms of words and improve the performance of a word embedding model. $word2vec_{LL}$model obtaining better results than $word2vec_{LR}$model proved this claim to be true. The third claim was made in Section **??**. There, we proposed that usage of lexical semantic similarity measures trained over a machine learning system can improve the overall system performance. The significant improvement that we show for the $word2vec_{LLS}$model over the $word2vec_{LL}$model verified this claim also to be accurate. Therefore we can conclude that the proposed methodology of word vector embedding augmented by lexical semantic similarity measures, gives a more accurate evaluation of the extent of which a given pair of words is semantically similar in the considered domain.

Semantic similarity measures are important in many areas of applications. Out of those, for future work, we expect to extend the findings of this study to the document level. Word based semantic similarity is the building block for sentence similarity measures, which in turn aggregates to build document similarity measures. This is the direction in which we intend to move. We will be using this word semantic similarity measure to build up to a document similarity measure which can be used for more efficient domain based document retrieval systems.

8.3. Deriving a Representative Vector for Ontology Classes with Instance Word Vector Embeddings

Conclusion and Future Works

In this work, we have demonstrated that the proposed method works as a better representation for a set of instances that occupy an ontology class than the traditional methods of using the average vector or the median vector. This discovery will be helpful in mainly two important tasks in the ontology domain.

The first one is further populating an already seeded ontology. We, in this study used the algorithm proposed in [20] for this task to obtain a test dataset. However, that approach has the weakness of being dependent on the WordNet lexicon. A methodology built on the representative vector discovery algorithm proposed in this work will not have that weakness. This is because all the necessary vector representations are obtained from word vector embeddings done using a corpus relevant to the given domain. Thus all the unique jargon would be adequately covered without much of a threat of conceptual drift. As future work, we expect to take this idea forward for automatic ontology population.

The second important task in the ontology domain that this method will be important is, class labeling. In this study we have demonstrated that our method is capable in deriving the closest vector representation to the class label. Thus, the converse of that would be true as well. That would be a *topic modeling* [155] task. The idea is that if given an unlabeled class, the method proposed by this study can be used to derive the representative vector. Then by querying the word embedding vector space it is possible to obtain the most suitable class label (topic) candidate.

8.4. Semi-Supervised Instance Population of an Ontology using Word Vector Embeddings

Conclusion and Future Works

Through this work we demonstrated the use of word embeddings on semi-supervised ontology population. We mainly focused on semi-supervised population which basically falls between the supervised population and unsupervised population. The main motive behind making the process semi-supervised is to reduce the level of manual interventions in ontology populations while maintaining a considerable amount of accuracy. As shown in the results, our ensemble model outperforms the five individual models in populating the selected legal ontology. The findings in this study is mainly important in two ways as mentioned below.

Firstly, an important part of the ontology engineering cycle is the ability to keep a handcrafted ontology up to date. Through the semi-supervised ontology population we can reduce the hassle involved in manual intervention to keep the ontology updated.

Secondly, there is novelty in the methodology proposed in our study. We proved that, since word embeddings map words or phrases from the vocabulary to vectors of real numbers based on the semantic context, a methodology based upon it can yield more sophisticated results when it comes to context sensitive tasks like ontology population. This indeed is a step up from the traditional information extraction based ontology population and maintenance processes, towards new horizons.

We can improve the methodology proposed, to yield better accuracy performances. For an example, we only considered the single word instances in populating the ontology using the defined models. However, in some of the scenarios, phrases also could be instances of ontology classes. Hence, it is important to convert phases to vectors and use them in the methodology as well. Also, as illustrated with models M_4 and M_5, we can perform more sophisticated semi-supervised ontology populations based on the concept of this study with more improvements. We keep them to be the future works of this study.

8.5. Legal Document Retrieval using Document Vector embeddings and Deep Neural Networks

Conclusion and Future Works

The hypothesis of this study was to prove the high accuracy levels of the $doc2vec_{NN}$ model against $doc2vec_{SSM}$ model and $doc2vec_{NV}$ model. The accuracy levels obtained by the Semantic Processing and Natural Language Processing techniques via the $doc2vec_{SSM}$ model, shows lesser values in terms of results over the $doc2vec_{NV}$ model, which was trained using Node2vec, where input was the Mention Map of this study. But the $doc2vec_{NN}$ model, which is an ensemble of the above two models, gives a significantly higher accuracy level as expected in our hypothesis.

We adopted semantic similarity measures from a previous study and generated a document to vector space to perform document retrieval tasks. This novel approach has shown better accuracy levels, as expected. However, we identified a practical limitation in carrying out this study which we intend to keep as the future work. The number of times a particular legal case is mentioned in a case was not taken into the account in this experiment. It should be noted that, a case which has been mentioned many times has a significant relevance to the case which the case was mentioned in, than a case which has mentioned few times. This could be taken into consideration by allocating a weight to each case based on the number of times it has been mentioned in a case.

The domain of interest in this study was the legal domain, and as mentioned, other domain areas like biology, astronomy, and geology, contain a similar syntactic structure within the domain corpora. The trained model built using our neural network, can be extended towards other domains to create similar information retrieval (IR) systems. Domain specific IR systems stand out in the field due to its complexities and difficulties. Our study has proven the importance of considering domain specific IR systems, which would indeed contribute towards the semantic web development [177].

Bibliography

[1] Daya C Wimalasuriya and Dejing Dou. Ontology-based information extraction: An introduction and a survey of current approaches. *Journal of Information Science*, 2010.

[2] Pepijn RS Visser and Trevor JM Bench-Capon. A comparison of four ontologies for the design of legal knowledge systems. *Artificial Intelligence and law*, 6(1):27–57, 1998.

[3] Ioan Alfred LETIA and Sorina CORNOIU. An ontological approach to legal literature for improving legal knowledge dissemination. *Development and Application Systems*, page 90, 2010.

[4] Stefanie Brüninghaus and Kevin D Ashley. Improving the representation of legal case texts with information extraction methods. In *Proceedings of the 8th international conference on Artificial intelligence and law*, pages 42–51. ACM, 2001.

[5] Denis Andrei de ARAUJO, RIGO Sandro José, Carolina Müller, and Rove Chishman. Information extraction from legal documents using linguistic knowledge and ontologies.

[6] John Stuart Mill. *On liberty*. Broadview Press, 1999.

[7] Diane E Oliver, Yuval Shahar, et al. Representation of change in controlled medical terminologies. *Artificial intelligence in medicine*, 15(1):53–76, 1999.

[8] Nisansa de Silva, Dejing Dou, and Jingshan Huang. Discovering inconsistencies in pubmed abstracts through ontology-based information extraction. In *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 362–371. ACM, 2017.

[9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[10] NHND de Silva, CSNJ Fernando, MKDT Maldeniya, et al. Semap-mapping dependency relationships into semantic frame relationships. In *17th ERU Research Symposium*, volume 17. Faculty of Engineering, University of Moratuwa, Sri Lanka, 2011.

[11] Hwee Tou Ng and Hian Beng Lee. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In *34th annual meeting on Association for Computational Linguistics*, pages 40–47, 1996.

[12] NHND de Silva. Safs3 algorithm: Frequency statistic and semantic similarity based semantic classification use case. *Advances in ICT for Emerging Regions (ICTer), 2015 Fifteenth International Conference on*, pages 77–83, 2015.

[13] Vindula Jayawardana, Dimuthu Lakmal, Nisansa de Silva, Amal Shehan Perera, Keet Sugathadasa, and Buddhi Ayesha. Deriving a representative vector for ontology classes with instance word vector embeddings. *arXiv preprint arXiv:1706.02909*, 2017.

[14] Vindula Jayawardana, Dimuthu Lakmal, Nisansa de Silva, Amal Shehan Perera, Keet Sugathadasa, Buddhi Ayesha, and Madhavi Perera. Semi-supervised instance population of an ontology using word vector embeddings. *arXiv preprint arXiv:1709.02911*, 2017.

[15] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, pages 133–138. Association for Computational Linguistics, 1994.

[16] Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc of 10th International Conference on Research in Computational Linguistics, ROCLING'97*, 1997.

[17] Graeme Hirst, David St-Onge, et al. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic lexical database*, 305:305–332, 1998.

[18] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition, 5(2):199-220*, 1993.

[19] Xi-Quan Yang, Na Sun, Tie-Li Sun, et al. The application of latent semantic indexing and ontology in text classification. *International Journal of Innovative Computing, Information and Control*, 5(12): 4491–4499, 2009.

[20] NHND De Silva, AS Perera, and MKDT Maldeniya. Semi-supervised algorithm for concept ontology based word set expansion. *Advances in ICT for Emerging Regions (ICTer), 2013 International Conference on*, pages 125–131, 2013.

[21] George A Miller, Richard Beckwith, Christiane Fellbaum, et al. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244, 1990.

[22] Indeewari Wijesiri, Malaka Gallage, Buddhika Gunathilaka, Madhuranga Lakjeewa, Daya C Wimalasuriya, Gihan Dias, Rohini Paranavithana, and Nisansa De Silva. Building a wordnet for sinhala. In *7th Global Wordnet Conference*, page 100, 2014.

[23] Jingshan Huang, Fernando Gutierrez, Harrison J Strachan, Dejing Dou, Weili Huang, Barry Smith, Judith A Blake, Karen Eilbeck, Darren A Natale, Yu Lin, et al. Omnisearch: a semantic search system based on the ontology for microrna target (omit) for microrna-target gene interaction data. *Journal of biomedical semantics*, 7(1):1, 2016.

[24] Jingshan Huang, Karen Eilbeck, Barry Smith, et al. The development of non-coding RNA ontology. *International journal of data mining and bioinformatics*, 15(3):214–232, 2016.

[25] Tomas Mikolov, Ilya Sutskever, et al. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[26] Nicola Guarino. Formal ontology and information systems. *Proceedings of FOIS'98, Trento, Italy*, 1998.

[27] George A Miller. Nouns in wordnet: a lexical inheritance system. *International journal of Lexicography*, 3(4):245–264, 1990.

[28] Christiane Fellbaum. *WordNet*. Wiley Online Library, 1998.

[29] Rosario Girardib Carla Fariaa, Ivo Serrab. A domain-independent process for automatic ontology population from text. *Science of Computer Programming*, 95:26—43, 2014.

[30] Juergen Rilling Rene Witte, Ninus Khamis. Flexible ontology population from text: The owlexporter.

[31] G Salton and ME Lesk. Iv information analysis and dictionary construction. 1971.

[32] Hans-Michael Müller, Eimear E Kenny, and Paul W Sternberg. Textpresso: an ontology-based information retrieval and extraction system for biological literature. *PLoS Biol*, 2(11):e309, 2004.

[33] William A Woods. Progress in natural language understanding: an application to lunar geology. In *Proceedings of the June 4-8, 1973, national computer conference and exposition*, pages 441–450. ACM, 1973.

[34] Meinard Müller. *Information retrieval for music and motion*, volume 2. Springer, 2007.

[35] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[36] JW Hughes. Rules for mediation in findlaw for legal professionals, 1999.

[37] Janet L Kolodner. An introduction to case-based reasoning. *Artificial intelligence review*, 6(1):3–34, 1992.

[38] Peter Jackson, Khalid Al-Kofahi, Chris Kreilick, and Brian Grom. Information extraction from case law and retrieval of prior cases by partial parsing and query generation. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 60–67. ACM, 1998.

[39] Adam Wyner and Wim Peters. On rule extraction from regulations. In *JURIX*, volume 11, pages 113–122, 2011.

[40] Carlo Biagioli, Enrico Francesconi, Andrea Passerini, Simonetta Montemagni, and Claudia Soria. Automatic semantics extraction in law documents. In *Proceedings of the 10th international conference on Artificial intelligence and law*, pages 133–140. ACM, 2005.

[41] Agnieszka Mykowiecka, Małgorzata Marciniak, and Anna Kupść. Rule-based information extraction from patients' clinical data. *Journal of biomedical informatics*, 42(5):923–936, 2009.

[42] Gang Chen, Baoran An, and Sifeng Zeng. A rule-based information extraction system for human-readable semi-structured scientific documents. In *Computer Science and Network Technology (ICCSNT), 2015 4th International Conference on*, volume 1, pages 75–84. IEEE, 2015.

[43] Frederick Reiss, Sriram Raghavan, Rajasekar Krishnamurthy, Huaiyu Zhu, and Shivakumar Vaithyanathan. An algebraic approach to rule-based information extraction. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 933–942. IEEE, 2008.

[44] Laura Chiticariu, Yunyao Li, and Frederick R Reiss. Rule-based information extraction is dead! long live rule-based information extraction systems! In *EMNLP*, number October, pages 827–832, 2013.

[45] Stephan Walter and Manfred Pinkal. Automatic extraction of definitions from german court decisions. In *Proceedings of the workshop on information extraction beyond the document*, pages 20–28. Association for Computational Linguistics, 2006.

[46] Guiraude Lame. Using nlp techniques to identify legal ontology components: Concepts and relations. *Artificial Intelligence and Law*, 12(4):379–396, 2004.

[47] José Saias and Paulo Quaresma. A methodology to create legal ontologies in a logic programming information retrieval system. In *Law and the Semantic Web*, pages 185–200. Springer, 2005.

[48] Paulo Quaresma and Teresa Gonçalves. Using linguistic information and machine learning techniques to identify entities from juridical documents. In *Semantic Processing of Legal Texts*, pages 44–59. Springer, 2010.

[49] Richard M Tong and Lee A Appelbaum. Machine learning for knowledge-based document routing (a report on the trec-2 experiment). *NIST SPECIAL PUBLICATION SP*, pages 253–253, 1994.

[50] Hinrich Schütze, David A Hull, and Jan O Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 229–237. ACM, 1995.

[51] Marko Grobelnik. Feature selection for unbalanced class distribution and naive bayes. ICML, 1999.

[52] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *ICML*, volume 99, pages 200–209, 1999.

[53] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1-4):43–52, 2010.

[54] Thorsten Joachims. Making large-scale svm learning practical. Technical report, Technical Report, SFB 475: Komplexitätsreduktion in Multivariaten Datenstrukturen, Universität Dortmund, 1998.

[55] Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation. In *Encyclopedia of database systems*, pages 532–538. Springer, 2009.

[56] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[57] Josef Sivic, Andrew Zisserman, et al. Video google: A text retrieval approach to object matching in videos. In *iccv*, volume 2, pages 1470–1477, 2003.

[58] Derek Greene and Pádraig Cunningham. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd international conference on Machine learning*, pages 377–384. ACM, 2006.

[59] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, 2003.

[60] Alessandro Perina, Nebojsa Jojic, Manuele Bicego, and Andrzej Truski. Documents as multiple overlapping windows into grids of counts. In *Advances in Neural Information Processing Systems*, pages 10–18, 2013.

[61] Ah-Hwee Tan et al. Text mining: The state of the art and the challenges. In *Proceedings of the PAKDD 1999 Workshop on Knowledge Disocovery from Advanced Databases*, volume 8, pages 65–70. sn, 1999.

[62] Peggy M Andersen, Philip J Hayes, Alison K Huettner, Linda M Schmandt, Irene B Nirenburg, and Steven P Weinstein. Automatic extraction of facts from press releases to generate news stories. In *Proceedings of the third conference on Applied natural language processing*, pages 170–177. Association for Computational Linguistics, 1992.

[63] Robert Gaizauskas and Yorick Wilks. Information extraction: Beyond document retrieval. *Journal of documentation*, 54(1):70–105, 1998.

[64] Ralph Grishman and Beth Sundheim. Message understanding conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, volume 1, 1996.

[65] Laurens Mommers, Wim Voermans, Wouter Koelewijn, and Hugo Kielman. Understanding the law: improving legal knowledge dissemination by translating the contents of formal sources of law. *Artificial Intelligence and Law*, 17(1):51–78, 2009. ISSN 1572-8382. doi: 10.1007/s10506-008-9073-5. URL `http://dx.doi.org/10.1007/s10506-008-9073-5`.

[66] Denis Andrei de Araujo, Carolina Müller, Rove Chishman, and Sandro José Rigo. Information extraction for legal knowledge representation–a review of approaches and trends. *Revista Brasileira de Computação Aplicada*, 6(2):2–19, 2014.

[67] Stuart Russell, Peter Norvig, and Artificial Intelligence. A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, 25:27, 1995.

[68] Ellen Riloff. Information extraction as a stepping stone toward story understanding. In *Understanding Language Understanding: Computational Models of Reading*, 1999.

[69] Peter Norvig Stuart Russell. Artificial intelligence: A modern approach (2nd edition). page 848–850, 2003.

[70] Jakub Piskorski and Roman Yangarber. *Information Extraction: Past, Present and Future*, pages 23–49. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-28569-1. doi: 10.1007/978-3-642-28569-1_2. URL `https://doi.org/10.1007/978-3-642-28569-1_2`.

[71] S. Fenz T. Neubauer J. Heurix, A. Rella. Automated transformation of semi-structured text elements. 2012.

[72] M. Rodrigues and A. Teixeira. *Advanced Applications of Natural Language Processing for Performing Information Extraction*. SpringerBriefs in Electrical and Computer Engineering. Springer International Publishing, 2015. ISBN 9783319155630. URL `https://books.google.lk/books?id=PKO1CQAAQBAJ`.

[73] Christopher Dozier and Robert Haschart. Automatic extraction and linking of person names in legal text. In *Content-Based Multimedia Information Access-Volume 2*, pages 1305–1321. Le Centre De Hautes Etudes Internales D'informatique Documentare, 2000.

[74] Maria Teresa Sagri Daniela Tiscornia. Legal concepts and multilingual contexts in digital information. *Beijing Law Review*, 3, 2012.

[75] Eneldo Loza Mencía. Segmentation of legal documents. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, ICAIL '09, pages 88–97, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-597-0. doi: 10.1145/1568234.1568245. URL `http://doi.acm.org/10.1145/1568234.1568245`.

[76] Christopher Manning. Information extraction and named entity recognition, 2012.

[77] Emmanuel Chieze, Atefeh Farzindar, and Guy Lapalme. An automatic system for summarization and information extraction of legal information. In *Semantic Processing of Legal Texts*, pages 216–234. Springer, 2010.

[78] Thomson Reuters. Westlaw, 1992. URL `https://www.westlaw.com/`.

[79] RELX Group. Lexisnexis, 2006. URL `https://www.lexisnexis.com/`.

[80] Atefeh Farzindar. *Résumé automatique de textes juridiques*. Université de Montréal, 2005.

[81] Jerry R Hobbs. Information extraction. In *Handbook of Natural Language Processing, Second Edition*, pages 511–532. Chapman and Hall/CRC, 2010.

[82] Rudi Studer, V Richard Benjamins, and Dieter Fensel. Knowledge engineering: principles and methods. *Data & knowledge engineering*, 25(1-2):161–197, 1998.

[83] Denis Andrei de Araujo, Sandro José Rigo, and Jorge Luis Victória Barbosa. Ontology-based information extraction for juridical events with case studies in brazilian legal realm. *Artificial Intelligence and Law*, Jul 2017. ISSN 1572-8382. doi: 10.1007/s10506-017-9203-z. URL `https://doi.org/10.1007/s10506-017-9203-z`.

[84] Andrew Stranieri and John Zeleznikow. *Knowledge discovery from legal databases*, volume 69. Springer Science & Business Media, 2006.

[85] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[86] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[87] Thorsten Joachims. *Learning to classify text using support vector machines: Methods, theory and algorithms*. Kluwer Academic Publishers, 2002.

[88] A. Passerini S.Montemagni C. Soria C. Biagioli, E. Francesconi. Automatic semantics extraction in law documents. In *ICAIL '05 Proceedings of the 10th international conference on Artificial intelligence and law*, pages 133–140. ACM, 2005.

[89] Paul B Laurent B, Daniële B. Extracting legal knowledge by means of a multilayer neural network application to municipal jurisprudence. In *ICAIL '91 Proceedings of the 3rd international conference on Artificial intelligence and law*, pages 288–296. ACM, 1991.

[90] Hsinchun C Michael C, Jennifer J. dg.o '02 proceedings of the 2002 annual national conference on digital government research. In *ICAIL '91 Proceedings of the 3rd international conference on Artificial intelligence and law*, pages 1–5. Digital Government Society of North America, 2002.

[91] William M Tianhao W. A semi-supervised active learning algorithm for information extraction from textual data. *Journal of the Association for Information Science and Technology*, 56:258–271, 2005.

[92] Robin Collier. Automatic template creation for information extraction. 1998.

[93] US Fast. A finite-state processor for information e x traction from real-word text. *Pro.*

[94] Tim Bray, Jean Paoli, C Michael Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible markup language (xml). *World Wide Web Journal*, 2(4):27–66, 1997.

[95] Silvio Peroni. *Semantic Web Technologies and Legal Scholarly Publishing*, volume 15. Springer, 2014.

[96] Fabio Vitali, L Bompani, and P Ciancarini. Hypertext functionalities with xml. *MARKUP LANGUAGES THEORY AND PRACTICE*, 2(4):389–410, 2000.

[97] Andrea Marchetti, Fabrizio Megale, Enrico Seta, and Fabio Vitali. Using xml as a means to access legislative documents: Italian and foreign experiences. *SIGAPP Appl. Comput. Rev.*, 10(1):54–62, April 2002. ISSN 1559-6915. doi: 10.1145/568235.568246. URL `http://doi.acm.org/10.1145/568235.568246`.

[98] Kincho H Gio W Gloria T, Shawn K. An e-government information architecture for regulation analysis and compliance assistance. In *6th International Conference on Electronic Commerce*, pages 461–470. ACM, 2004.

[99] M Mercedes Martínez, Pablo de la Fuente, and Jean-Claude Derniame. Xml as a means to support information extraction from legal documents. *Comput. Syst. Sci. Eng.*, 18(5):263–277, 2003.

[100] Shlomo Argamon, Ido Dagan, and Yuval Krymolowski. A memory-based approach to learning shallow natural language patterns. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*, volume 1, 1998.

[101] Keet Sugathadasa, Buddhi Ayesha, Nisansa de Silva, Amal Shehan Perera, Vindula Jayawardana, Dimuthu Lakmal, and Madhavi Perera. Synergistic union of word2vec and lexicon for domain specific semantic similarity. *arXiv preprint arXiv:1706.01967*, 2017.

[102] James Allen, Mehdi Manshadi, Myroslava Dzikovska, and Mary Swift. Deep linguistic processing for spoken dialogue systems. In *Proceedings of the Workshop on Deep Linguistic Processing*, pages 49–56. Association for Computational Linguistics, 2007.

[103] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.

[104] Hamish Cunningham. Gate, a general architecture for text engineering. *Computers and the Humanities*, 36(2):223–254, 2002.

[105] Kalina Bontcheva, Marin Dimitrov, Diana Maynard, Valentin Tablan, and Hamish Cunningham. Shallow methods for named entity coreference resolution. In *Chaınes de références et résolveurs d'anaphores, workshop TALN*, 2002.

[106] Lawrence Reeve and Hyoil Han. Survey of semantic annotation platforms. In *Proceedings of the 2005 ACM symposium on Applied computing*, pages 1634–1638. ACM, 2005.

[107] Paul A Kogut and William S Holmes III. Aerodaml: Applying information extraction to generate daml annotations from web pages. In *Semannot@ K-CAP 2001*, 2001.

[108] Paul Kogut, Stephen Cranefield, Lewis Hart, Mark Dutra, Kenneth Baclawski, Mieczyslaw Kokar, and Jeffrey Smith. Uml for ontology development. *The Knowledge Engineering Review*, 17(1):61–64, 2002.

[109] Fabio Ciravegna, Sam Chapman, Alexiei Dingli, and Yorick Wilks. Learning to harvest information for the semantic web. *The Semantic Web: Research and Applications*, pages 312–326, 2004.

[110] Maria Vargas-Vera, Enrico Motta, John Domingue, Mattia Lanzoni, Arthur Stutt, and Fabio Ciravegna. Mnm: Ontology driven semi-automatic and automatic support for semantic markup. *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, pages 213–221, 2002.

[111] Diana Maynard. Multi-source and multilingual information extraction. *Expert Update*, 6(3):11–16, 2003.

[112] Siegfried Handschuh, Steffen Staab, and Fabio Ciravegna. S-cream—semi-automatic creation of metadata. *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, pages 165–184, 2002.

[113] Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, Ramanathan Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A Tomlin, et al. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th international conference on World Wide Web*, pages 178–186. ACM, 2003.

[114] Martin Myhill. Handbook of research on digital libraries: Design, development and impact. *Program*, 2013.

[115] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[116] Piek Vossen. Introduction to eurowordnet. *Computers and the Humanities*, 32(2-3):73–89, 1998.

[117] Birgit Hamp, Helmut Feldweg, et al. Germanet-a lexical-semantic net for german. In *Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 9–15, 1997.

[118] Jordi Atserias, Luıs Villarejo, German Rigau, Eneko Agirre, John Carroll, Bernardo Magnini, and Piek Vossen. The meaning multilingual central repository. 2004.

[119] Adam Pease, Christiane Fellbaum, and Piek Vossen. Building the global wordnet grid. *CIL18*, 2008.

[120] Stephen D Richardson, William B Dolan, and Lucy Vanderwende. Mindnet: acquiring and structuring semantic information from text. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 2*, pages 1098–1102. Association for Computational Linguistics, 1998.

[121] Manish Sinha, Mahesh Reddy, and Pushpak Bhattacharyya. An approach towards construction and application of multilingual indo-wordnet. In *3rd Global Wordnet Conference (GWC 06), Jeju Island, Korea*, 2006.

[122] Jeffrey EF Friedl. *Mastering regular expressions.* " O'Reilly Media, Inc.", 2002.

[123] Soraya Zaidi, MT Laskri, and Ahmed Abdelali. Arabic collocations extraction using gate. In *Machine and Web Intelligence (ICMWI), 2010 International Conference on*, pages 473–475. IEEE, 2010.

[124] Diana Maynard, Milena Yankova, Alexandros Kourakis, and Antonis Kokossis. Ontology-based information extraction for market monitoring and technology watch. In *ESWC Workshop" End User Apects of the Semantic Web"), Heraklion, Crete*, 2005.

[125] Adam Z Wyner and Wim Peters. Lexical semantics and expert legal knowledge towards the identification of legal case factors. In *JURIX*, volume 10, pages 127–136, 2010.

[126] Adam Z Wyner. Towards annotating and extracting textual legal case elements. *Informatica e Diritto: special issue on legal ontologies and artificial intelligent techniques*, 19(1-2):9–18, 2010.

[127] Atefeh Farzindar and Guy Lapalme. Legal text summarization by exploration of the thematic structures and argumentative roles. In *Text summarization branches out workshop held in conjunction with ACL*, pages 27–34, 2004.

[128] Flora Amato, Antonino Mazzeo, Vincenzo Moscato, and Antonio Picariello. A system for semantic retrieval and long-term preservation of multimedia documents in the e-government domain. *International Journal of Web and Grid Services*, 5(4):323–338, 2009.

[129] Pete Houston. *Instant jsoup How-to.* Packt Publishing Ltd, 2013.

[130] Leonard Richardson. Beautiful soup documentation, 2007.

[131] Yuhong Cai, John Grundy, and John Hosking. Synthesizing client load models for performance engineering via web crawling. In *Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering*, pages 353–362. ACM, 2007.

[132] G. V. Cormack M. R. Grossman. The grossman-cormack glossary of technology-assisted review with foreword by john m. facciola, u.s. magistrate judge. In *Federal Courts Law Review*, 2013.

[133] John Makhoul, Francis Kubala, Richard Schwartz, and Ralph Weischedel. Performance measures for information extraction. In *In Proceedings of DARPA Broadcast News Workshop*, pages 249–252, 1999.

[134] Nisansa de Silva, Danaja Maldeniya, and Chamilka Wijeratne. Subject specific stream classification preprocessing algorithm for twitter data stream. *arXiv preprint arXiv:1705.09995*, 2017.

[135] Joseph John Rocchio. Relevance feedback in information retrieval. 1971.

[136] Gerard Salton, Amit Singhal, Mandar Mitra, and Chris Buckley. Automatic text structuring and summarization. *Information Processing & Management*, 33(2):193–207, 1997.

[137] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[138] Mirella Lapata and Regina Barzilay. Automatic evaluation of text coherence: Models and representations. In *IJCAI*, volume 5, pages 1085–1090, 2005.

[139] Egidio Terra and Charles LA Clarke. Frequency estimates for statistical word similarity measures. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 165–172. Association for Computational Linguistics, 2003.

[140] Rada Mihalcea, Courtney Corley, Carlo Strapparava, et al. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, volume 6, pages 775–780, 2006.

[141] Gerard Salton and Michael J McGill. Introduction to modern information retrieval. 1986.

[142] Richard W Hamming. Error detecting and error correcting codes. *Bell Labs Technical Journal*, 29(2): 147–160, 1950.

[143] Mohammad Norouzi, David J Fleet, and Ruslan R Salakhutdinov. Hamming distance metric learning. In *Advances in neural information processing systems*, pages 1061–1069, 2012.

[144] Sung-Hyuk Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007.

[145] Sung-Hyuk Cha, Sungsoon Yoon, and Charles C Tappert. Enhancing binary feature vector similarity measures. 2005.

[146] Gang Qian, Shamik Sural, Yuelong Gu, and Sakti Pramanik. Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1232–1237. ACM, 2004.

[147] David A Evans, James J Cimino, William R Hersh, Stanley M Huff, Douglas S Bell, et al. Toward a medical-concept representation language. *Journal of the American Medical Informatics Association*, 1(3):207–217, 1994.

[148] Buzhou Tang, Hongxin Cao, Yonghui Wu, Min Jiang, and Hua Xu. Recognizing clinical entities in hospital discharge summaries using structural support vector machines with word representation features. *BMC medical informatics and decision making*, 13(1):S1, 2013.

[149] Erich Schweighofer and Werner Winiwarter. Legal expert system konterm—automatic representation of document structure and contents. In *International Conference on Database and Expert Systems Applications*, pages 486–497. Springer, 1993.

[150] John J Nay. Gov2vec: Learning distributed representations of institutions and their legal text. *arXiv preprint arXiv:1609.06616*, 2016.

[151] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[152] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into texts. Association for Computational Linguistics, 2004.

[153] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[154] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[155] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian lda for topic models with word embeddings. In *ACL (1)*, pages 795–804, 2015.

[156] Oren Melamud, Omer Levy, et al. A simple word embedding model for lexical substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, 2015.

[157] Adam Funk Diana Maynard and Wim Peters. Sprat: a tool for automatic semantic pattern-based ontology population. 2009.

[158] Andrew Carlson, Justin Betteridge, Richard C. Wang, E. R. Hruschka, Jr., and Tom M. Mitchell. Coupled semi-supervised learning for information extraction. *WSDM '10 Proceedings of the third ACM international conference on Web search and data mining*, pages 101–110, 2010.

[159] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, E. R. Hruschka, Jr., and Tom M. Mitchell. Toward an architecture for never-ending language learning. *Proceeding AAAI'10 Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 1306–1313, 2010.

[160] Ruslan Salakhutdinov Zhilin Yang, William W. Cohen. Revisiting semi-supervised learning with graph embeddings. *Proceedings of International Conference on Machine Learning*, 2016.

[161] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[162] Yassine Drias. Towards a model of storage oriented nosql for an ontology database. 2014.

[163] Tuomo Korenius, Jorma Laurikkala, Kalervo Järvelin, and Martti Juhola. Stemming and lemmatization in the clustering of finnish text documents. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 625–633. ACM, 2004.

[164] Edgar Osuna, Robert Freund, and Federico Girosi. An improved training algorithm for support vector machines. pages 276–285, 1997.

[165] FindLaw cases and codes. `http://caselaw.findlaw.com/`. Accessed: 2017-05-18.

[166] Dragomir R Radev. A common theory of information fusion from multiple text sources step one: cross-document structure. In *Proceedings of the 1st SIGdial workshop on Discourse and dialogue-Volume 10*, pages 74–83. Association for Computational Linguistics, 2000.

[167] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196, 2014.

[168] Xingxing Wang, LiMin Wang, and Yu Qiao. A comparative study of encoding, pooling and normalization methods for action recognition. In *Asian Conference on Computer Vision*, pages 572–585. Springer, 2012.

[169] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

[170] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Stanford, CA, 1995.

[171] Dieter Fensel. *Spinning the Semantic Web: bringing the World Wide Web to its full potential.* Mit Press, 2005.

[172] W3c semantic web activity. 2011. Retrieved November 26, 2011.

[173] Pompeu Casanovas, Monica Palmirani, Silvio Peroni, Tom van Engers, and Fabio Vitali. Semantic web for the legal domain: the next step. *Semantic Web*, 7(3):213–227, 2016.

[174] Thomas R Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5-6):907–928, 1995.

[175] Mírian Bruckschen, Caio Northfleet, DM Silva, Paulo Bridi, Roger Granada, Renata Vieira, Prasad Rao, and Tomas Sander. Named entity recognition in the legal domain for ontology population. In *Workshop Programme*, page 16, 2010.

[176] David B Leake. *Case-based reasoning.* John Wiley and Sons Ltd., 2003.

[177] Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic web. *Scientific american*, 284(5): 28–37, 2001.