2016

A prototype for information capture using natural language processing: a case of Nation online news

Mbuthia, S. K.
Faculty of Information Technology (FIT)
Strathmore University

A PROTOTYPE FOR INFORMATION CAPTURE USING NATURAL LANGUAGE PROCESSING: A CASE OF NATION ONLINE NEWS

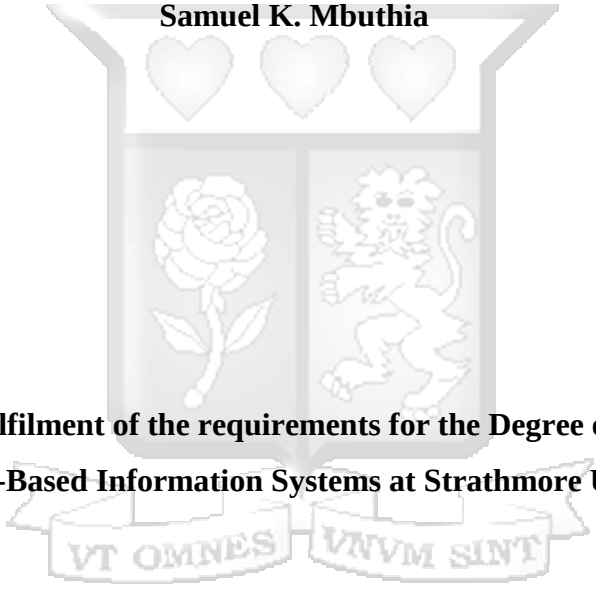
Samuel K. Mbuthia

Master of Science in Computer-Based Information Systems

April, 2016

A PROTOTYPE FOR INFORMATION CAPTURE USING NATURAL LANGUAGE PROCESSING: A CASE OF NATION ONLINE NEWS

Samuel K. Mbuthia



Submitted in partial fulfilment of the requirements for the Degree of Master of Science in Computer-Based Information Systems at Strathmore University

Faculty of Information Technology

Strathmore University

Nairobi, Kenya

Declaration

I declare that this work has not been previously submitted and approved for the reward of a degree by this or any other University. To the best of my knowledge and belief, the dissertation contains no material previously published or written by another person except where due reference is made in the dissertation itself.

Student's Name : Samuel Kamau Mbuthia

Student Number : 083160

Signature : _____

Date : 08/06/2016

Approval

The dissertation of Samuel Kamau Mbuthia was reviewed and approved by the following:

Professor Ismail Ateya

Faculty of Information Technology

Strathmore University

Dr. Joseph Orero, PhD

Dean, Faculty of Information Technology

Strathmore University

Professor Ruth Kiraka

Dean, School of Graduate Studies

Strathmore University

Abstract

Natural Language Processing is a crucial component in the study of artificial intelligence. The Internet contains an abundance of information from a multitude of sources and a lot of this information is in human language which is not easily interpretable by computers. This is especially the case with unstructured data. Newswire sources, which include news websites and other forms of digital news reporting, provide easily accessible sources of information. This information is reliable because in many cases it is provided by journalists who are governed by professional codes of conduct. Years of reporting news on the Internet has resulted in a lot of information that is free and readily available. However, this information cannot be utilized effectively as it is too much to be processed by usual data mining techniques. This is where Natural Language Processing comes in as an essential tool in mining and processing the plethora of newswire content. This research took an applied approach to try and address the problem of inefficient utilization of newswire content. The research went through several stages each trying to address a specific area of the problem. While the problem being addressed was a global one, a focus on Kenyan news made the tasks manageable and specific but at the same time scalable to a larger setting. The research delved into the creation of a novel prototype that addressed the gaps in currently existing systems and improved on how information processed by Natural Language Processing techniques could be presented and reported. The application of graphical representations by using a number of advanced graphical libraries and application program interfaces as well as the application of machine learning models and algorithms were an integral part of the research. Various tests were undertaken to determine the viability of the prototype and key among them were performance test that monitored the performance of the applied machine learning algorithms. The results of the tests were duly tabulated and reported. While the research was largely successful, it was concluded that a lot more remains to be done to comprehensively address the gaps that exists in the area of research. As such, this research acts as a platform for further research that can gradually fill all the gaps identified as well as those that may be identified in the future.

Dedication

I dedicate this to all those who have assisted me to get to this point; particularly my parents Dr. Geoffrey Mbuthia and Mrs. Lilian Mbuthia; my sisters, Millicent Muthoni, Ruth Wairimu, and Julia Wambui. I also appreciate all my lecturers, friends, and fellow students who have been greatly supportive and instrumental in my endeavors so far.

Acknowledgments

I would like to acknowledge all those who have made this work possible; my supervisor Professor Ismail Ateya who ably guided me through this research, my lecturers who guided me in my academic work, my close friends Olive Murage and Susan Wamae who helped me mold my sketchy thoughts into sensible ideas and my employer Nation Media Group for providing me the leeway to study and conduct this research. Above all, I thank God for giving me the strength throughout this journey.

Abbreviations/Acronyms

API	–	Application Program Interface
BST	–	Binary Search Tree
EMM	–	Europe Media Monitor
HTML	–	Hypertext Markup Language
IDEA	–	Integrated Data for Events Analysis
JRC	–	European's Commission Joint Research Centre
JSON	–	JavaScript Object Notation
KEDS	–	Kansas Event Data System
LDA	–	Latent Dirichlet Allocation
MALLET	–	Machine Language for Learning Toolkit
NER	–	Named Entity Recognition
NLP	–	Natural Language Processing
NLTK	–	Natural Language Toolkit
PANDA	–	Protocol on Nonviolent Direct Action
PETRARCH	–	Python Engine for Text Resolution and Related Coding Hierarchy
POS	–	Part of Speech
RSS	–	Rich Site Summary
TABARI	–	Textual Analysis by Augmented Replacement Instructions
UML	–	Unified Modelling Language
URL	–	Uniform Resource Locator

VRA® – Virtual Research Associates

WBS – Work Breakdown Structure

WEIS – World Event/Interaction Survey

XML – Extensible Markup Language

Definition of Terms

Corpus - Any collection of language material (Dale, Moisl, & Somers, 2000).

Coreference Resolution - The task of finding all expressions that refer to the same entity in a text (Coreference Resolution, n.d.).

Lemma - The canonical or morphological form of a word (Perkins, 2010).

Lexicon - The vocabulary of a person, language, or branch of knowledge (Jackson & Moulinier, 2007).

Morphology - Knowledge of the meaningful components of words (Jurafsky & Martin, 2009).

Newswire - An electronically transmitted service providing news stories (Jurafsky & Martin, 2009).

Pragmatics - Knowledge of the relationship of meaning to the goals and intentions of the speaker (Jurafsky & Martin, 2009).

Syntax - Knowledge of the structural relationship between words (Jurafsky & Martin, 2009).

Utterance - An uninterrupted chain of spoken or written language (Jurafsky & Martin, 2009).

Table of Contents

List of Figures

List of Tables

List of Equations

Chapter 1: Introduction

1.1. Background of the Study

The ubiquity of the Internet has resulted in a source of information that is diverse, multilingual, multicultural and multidisciplinary. Online news has become a reliable source of information concerning various issues from disaster related events to security threats and situational monitoring during elections. Online news is considered by security authorities and organizations around the globe as an important source of information that can be exploited for early detection of certain threats and for situation monitoring during crises (Piskorski & Atkinson, 2011).

An ever-growing amount of information transmitted through the Internet has led to an emergence of advanced tools that combine techniques from text mining, machine learning, statistical analysis, and computational linguistics to help intelligence experts to manage the overflow of information, filter out the relevant from irrelevant, and to extract valuable and actionable knowledge from online sources (Piskorski et al., 2011). Projects such as the Kansas Event Data System (KEDS) have over long periods been using event data and other web-based sources to produce reports on International conflict and political events (Schrodt, Gerner, & Yilmaz, 2004). Such projects have been instrumental in the implementation of early warning systems and intelligence gathering.

Some of the systems in these projects however require a lot of human input; for example KEDS relies a lot on human coding of events. This approach is unsustainable in instances where funding is not available to pay the human event coders who do the work. For this reason, tools and software that use natural language processing techniques to extract event data have made it possible for information and event capture systems to be developed robustly. Among these are the Stanford CoreNLP software developed at Stanford University; Machine Language for Learning Toolkit (MALLET) developed at the University of Massachusetts, Amherst (McCallum, 2002); as well as Natural Language Processing Toolkit (NLTK) for Python (Perkins, 2010). These tools rely on algorithms based on linguistic rules to automatically process numerous text sources and extract desired information depending on the requirements of the processor. Using these tools, models have been created based on the previously existing event

extraction and human coding systems. These models have been able to harness the robust processing capabilities of computers to process even more text sources and within a much shorter time compared to previous models.

Piskorski, Tanev, Atkinson and Van Der Goot (2008) report on the multilingual event-extraction system developed at the Joint Research Centre of the European Commission for extracting violent and natural disaster event information from online news articles collected through the Internet with the Europe Media Monitor (EMM). The EMM news brief system is able to capture information on events, victims and entities that is extracted from news articles around the world (Piskorski et al., 2008). This information is then presented in graphs and maps. This kind of presentation makes it possible to analyze events from a very wide array of news sources and this highly improves the utilization of news articles. Through this, deeper analysis of information that has accumulated over time becomes possible and this could find applications in areas such as machine learning, early detection of threats, situation monitoring, and possibly trend forecasting. This kind of actionable intelligence is useful to many organizations and agencies, from security agencies that are trying to prevent escalation of security threats to government agencies trying to prevent economic crises in a country (Piskorski et al., 2008). Such intelligence could also be used by researchers to further advancements in the areas of machine learning, artificial intelligence, data mining and information extraction.

1.2. Problem Statement

The increasing amount of digital data and their sources has made utilization of extracted information in decision making processes difficult. Most of this data is initially unstructured and described using natural, human-understandable language making it limited in the degree in which it is machine-interpretable (Hogenboom, Fransicar, Kaymak & De Jong, 2011). Online news articles provide very crucial information which is very important over a short period. However, over a long period, this information becomes too much to utilize it comprehensively. This leads to information overflow. This research intends to solve this problem by creating prototype that will effectively capture information over a long period and novel methods and techniques to analyze and utilize it.

1.3. Research Objectives

i. To identify information that is useful in the utilization of news media content.
ii. To appraise the existing information capture tools and techniques and their utilization.
iii. To review the challenges existing in current information capture techniques and their utilization.
iv. To develop a prototype that will use novel techniques to utilize information extracted using Natural Language Processing.
v. To test the prototype.

1.4. Research Questions

i. What kind of information is most useful in utilization of news media content?
ii. What are the merits of the existing information capture and utilization techniques?
iii. What are the challenges faced by the current information capture techniques and their utilization?
iv. How can information be extracted and utilized efficiently and effectively?
v. What are the merits of the developed prototype?

1.5. Justification

While a lot of research has already been done in the area, there are gaps that still exist in how best extracted information can be utilized over a long period. This research aims at addressing the gaps by proposing a prototype that will use novel analytic and visualization techniques to improve utilization of extracted information. The proposed prototype will use previously researched methods in natural language processing to extract event information from online news. The prototype will save time for researchers who want to obtain intelligence from news media content. It will also be helpful to media organizations that want to analyze their news reporting trends. Government and non-governmental organizations will be able to analyze and investigate the protracted effects of disaster events that have been reported and how they relate to other occurrences and events. Finally, this research will act as a launch pad for deeper research into tools and techniques that can complement event information extraction using natural language processing. This research will add to the field of knowledge in the sphere of information extraction, natural language processing, and data analysis and visualization.

1.6. Scope and Limitations of the Study

This research will be focused only on major event information reported in the Daily Nation website between the years 2015 and 2016. This research is not aimed at creating new information extraction methods, but rather, use existing natural language processing methods and tools to generate information banks that will then be utilized using novel techniques and methods.

Chapter 2: Literature Review

2.1. Introduction

Language Processing has been a focus of research ever since the 1950's when machine translation was still a relatively new domain. Computer scientists and linguists have over the years developed frameworks and tools that would help simplify NLP and make it more efficient. Since the 1950's, work in NLP has attempted to use symbolic methods – where knowledge about language is explicitly encoded in rules or other forms of representation as a means of solving the problem of how to automatically process human languages (Dale et al., 2000). Language processing applications can be distinguished from other data processing systems by their use of knowledge of language (Jurafsky & Martin, 2009). Language processing makes use of the knowledge of language which encompasses an understanding of areas such as sentence structure, word morphology, and semantics. In language processing that involved spoken utterances in audio form, knowledge about phonetics and phonology is also required (Jurafsky & Martin, 2009).

2.2. Information Extraction using Natural Language Processing

2.2.1. Overview of Natural Language Processing.

NLP is the function of software or hardware components in a computer system which analyzes or synthesizes spoken or written language (Jackson & Moulinier, 2007). Language processing has an important role to play in both the production and packaging of online information. NLP has many applications in the real world today. Possible applications of NLP are finding names of people and companies in free texts and linking the names to public records or to a directory of companies (Jackson & Moulinier, 2007). NLP as a field within Artificial Intelligence and Machine Learning has been instrumental in ameliorating direct interaction of humans with machines. With further advancement in NLP machines will be able to learn directly from human language without further human input either through programming code or structured rules of interpretation.

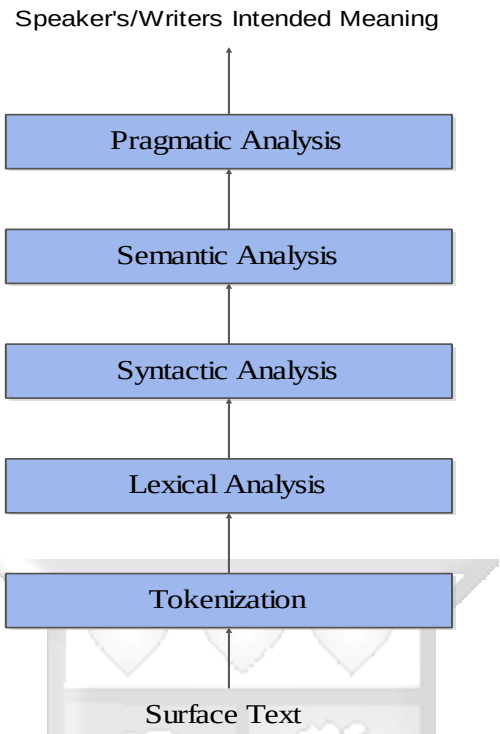


Figure 2.1: Stages of analysis in processing natural language (Dale et al., 2000).

Figure 2.1 demonstrates the various stages of analysis in processing natural language where the surface text represent the raw input from a news source or other text that is to be processed. This text is taken through the various processes from tokenization where the text is split into desirable segments or tokens. The subsequent processes then analyze the tokens until the intended meaning of the input text is discovered. In an event extraction system, the final output will be the event data which will include; type of event, actor, victims, victim count, and location.

2.2.2. Specialties and Sub-Specialties in Natural Language Processing.

According to Jurafsky (n.d.) Language Technology in Natural Language Processing can be categorized into specialties and sub-specialties which are in various stages of being solved as illustrated in Figure 2.2. Each of these specialties and sub-specialties utilizes particular aspects of linguistics. Sentence structure has been particularly important in named part of speech tagging and named entity recognition.

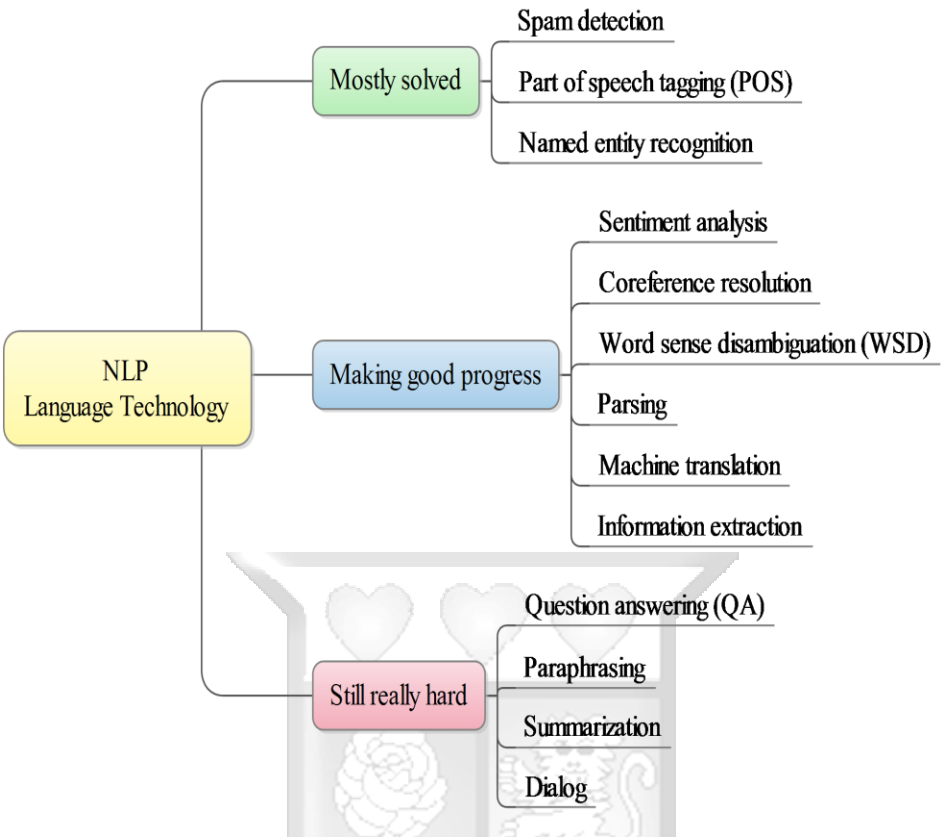


Figure 2.2: Specialties and subspecialties in NLP.

2.2.3. Stages of Analysis in Natural Language Processing.

A number of different techniques in NLP have been developed over the years. Many of these techniques are usually combined to be able to extract event data given information in free text such as online news articles. From Figure 2.1, some of the mentioned stages are:

2.2.3.1. Tokenization.

In NLP it is important to distinguish what constitutes a word and what constitutes a sentence (Jackson & Moulinier, 2007). Tokenization is the subdividing a text article into meaningful units; words, sentences, and paragraphs. Tokenization involves breaking up the sequence of character in a text by locating word boundaries (Jackson & Moulinier, 2007). These boundaries could be white space for words, period for sentences or escape character for paragraphs (Perkins J., 2010). Some of the challenges that arise in tokenization are due to language dependence, character-set dependence, application dependence, and corpus dependence

(Jackson & Moulinier, 2007). Different systems have adopted various methods to address these challenges as will be discussed later on in this chapter.

2.2.3.2. Lexical analysis.

Lexical analysis is broadly defined as the determination of lexical features for the words of a text (Jackson & Moulinier, 2007). The precise sets of features that are identified are largely dependent on the broader application within which the lexical analysis component is functioning (Dale et al., 2000). Lexical analyses vary with different languages. This research is however concerned specifically with lexical analysis in English.

2.2.3.3. Syntactic analysis.

Text is analyzed in terms of its syntax to provide an order and structure that is more amenable to an analysis in terms of semantics, or literal meaning (Dale et al., 2000). This is crucial especially in a multilingual system because of the difference in syntax among different languages. Syntactic analysis has been a key process in developing NLP systems that analyze content in languages like Chinese where the syntax is considerably different from that of English and other Indo-European languages.

2.2.3.4. Semantic analysis.

Understanding an utterance is a key goal in NLP systems. This may involve incorporating the information provided by the utterance into one's own knowledge base or performing some action in response to it (Dale et al., 2000).

2.2.3.5. Pragmatic analysis.

In this analysis, the meaning of the utterance or text in context is determined (Dale et al., 2000). This is however not an easy task because computer systems lack the human capability to understand the specific context of something that is being said. An example is the ability to interpret a serious statement from a statement that is made sarcastically.

2.2.4. Natural Language Processing Tools.

2.2.4.1. Stanford CoreNLP.

The Stanford CoreNLP is a Java based annotation pipeline framework which provides the common NLP steps, from tokenization through to coreference resolution (Manning, Surdeanu, Bauer, Finkel, Bethard & McClosky, 2014). The CoreNLP framework is open-source and simple to use for a wide range of requirements. Stanford CoreNLP provides a command-line interface and the ability to write out an annotation in various formats including XML (Manning et al., 2014). Figure 2.3 illustrates the system architecture for Stanford CoreNLP: Raw text is put into an Annotation object and then a sequence of annotators adds information in an analysis pipeline resulting in an output containing all the analysis information added by the annotators (Manning et al., 2014).

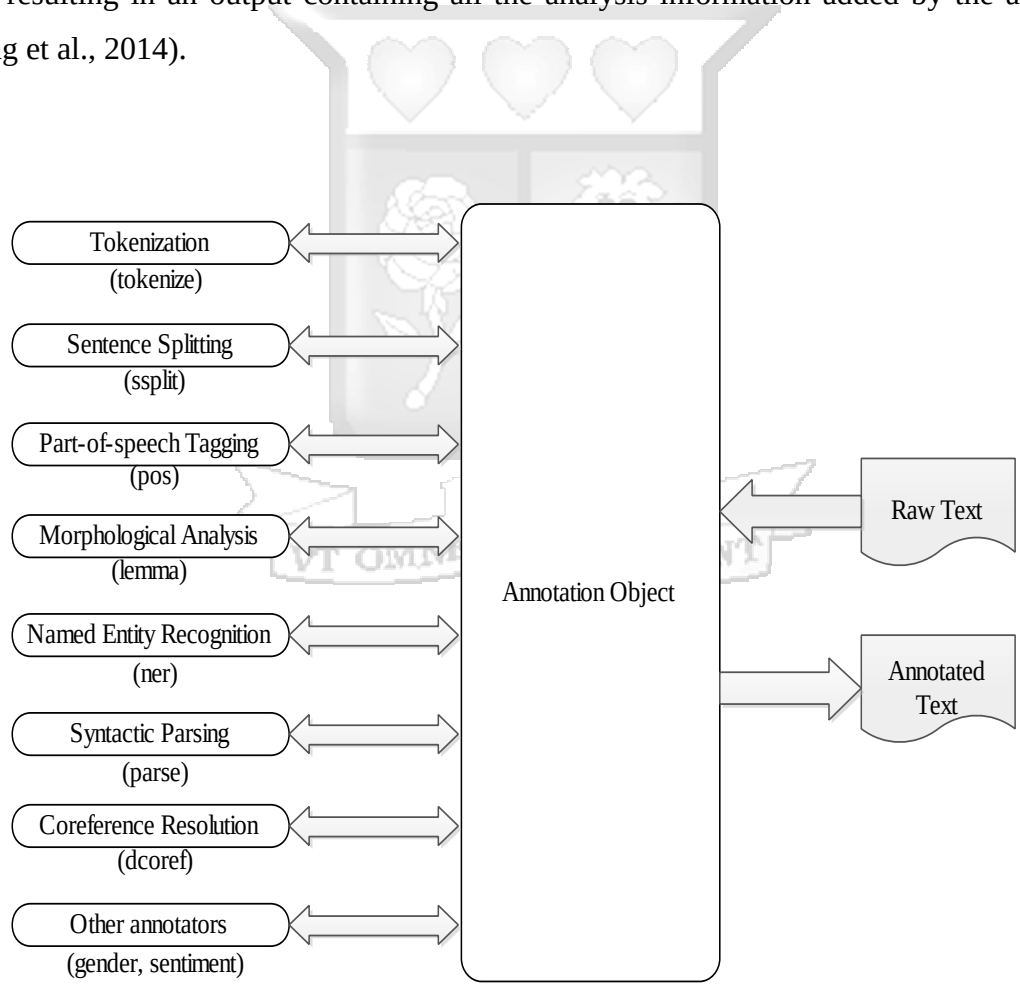


Figure 2.3: System architecture for the Stanford CoreNLP system (Manning et al., 2014).

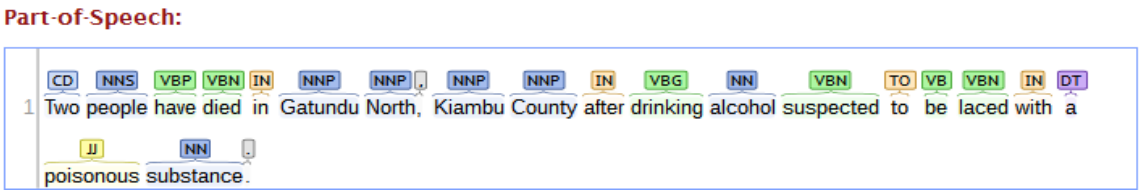


Figure 2.4: Visualization of POS tagging using Stanford CoreNLP.

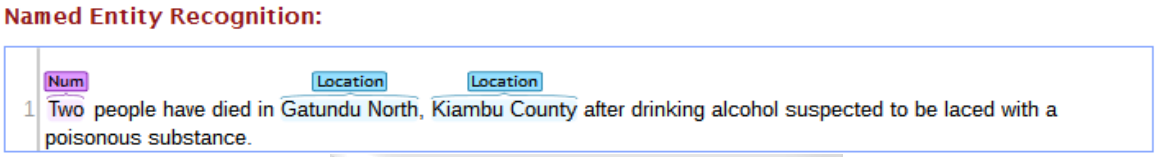


Figure 2.5: Visualization of NER using Stanford CoreNLP.

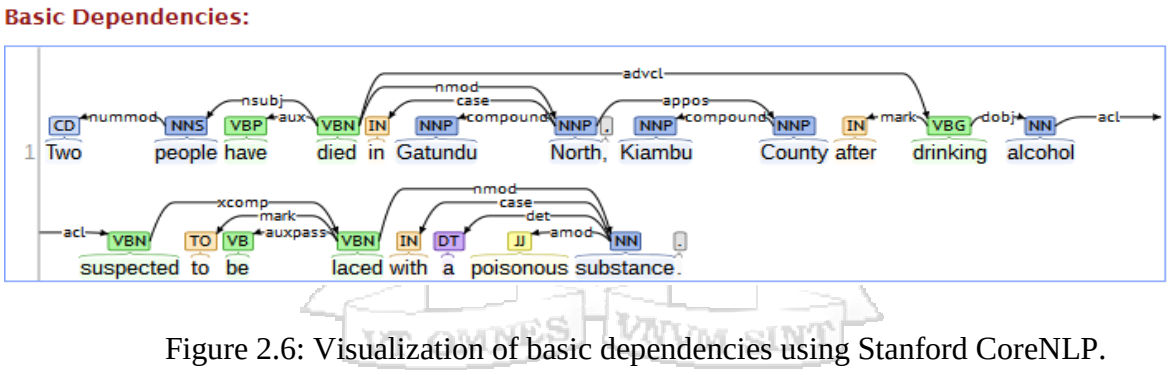


Figure 2.6: Visualization of basic dependencies using Stanford CoreNLP.

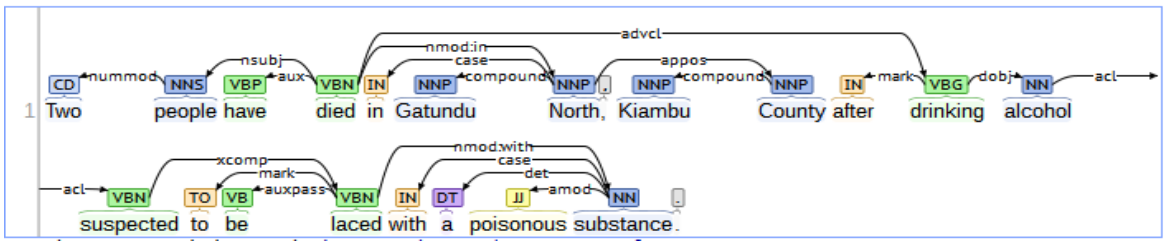


Figure 2.7: Visualization of enhanced dependencies using Stanford CoreNLP.

2.2.4.2. Machine Language for Learning Toolkit.

MALLET is an open source NLP software developed at the University of Massachusetts Amherst. It is a Java-based package for statistical natural language processing, document classification, clustering, topic modelling, information extraction, and other machine learning applications to text (McCallum, 2002). MALLET includes tools for document classification: efficient routines for converting text to features with algorithms including; Naïve Bayes, Maximum Entropy, and decision trees as well as code for evaluating classifier performance using several commonly used metrics (McCallum, 2002). MALLET also includes tools for sequence tagging for applications such as named-entity extraction from text with algorithms such as; Hidden Markov Models, Maximum Entropy Markov Models and Conditional Random Fields (McCallum, 2002). Other features included in MALLET are the topic modeling toolkit which contains efficient sampling-based implementations of Latent Dirichlet Allocation, Pachinko Allocation, and Hierarchical LDA (McCallum, 2002). Despite offering many of the features that are desirable in a natural language processing tool, more research needs to be done to determine the preference of MALLET as the NLP tool of choice at the expense of the better documented and more researched on Stanford CoreNLP.

2.2.4.3. Natural Language Toolkit for Python.

NLTK for Python is a platform for building Python programs to work with human language data. It provides interfaces to corpora and lexical resources as well as a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning (Bird, Klein, & Loper, 2009). NLTK was originally designed for teaching and has been adopted in the industry for research and development due to its usefulness and breadth of coverage (Perkins, 2010). Figure 2.8 illustrates the architecture for a simple information processing system which begins by processing a document using several procedures: the raw text of the document is first split into sentences using a sentence segmenter, each sentence is further subdivided into words using a tokenizer and then each sentence is tagged with part-of-speech tags which are then applied in named entity detection, and finally, relation detection is used to search for likely relations between different entities in the text (Bird et al., 2009)

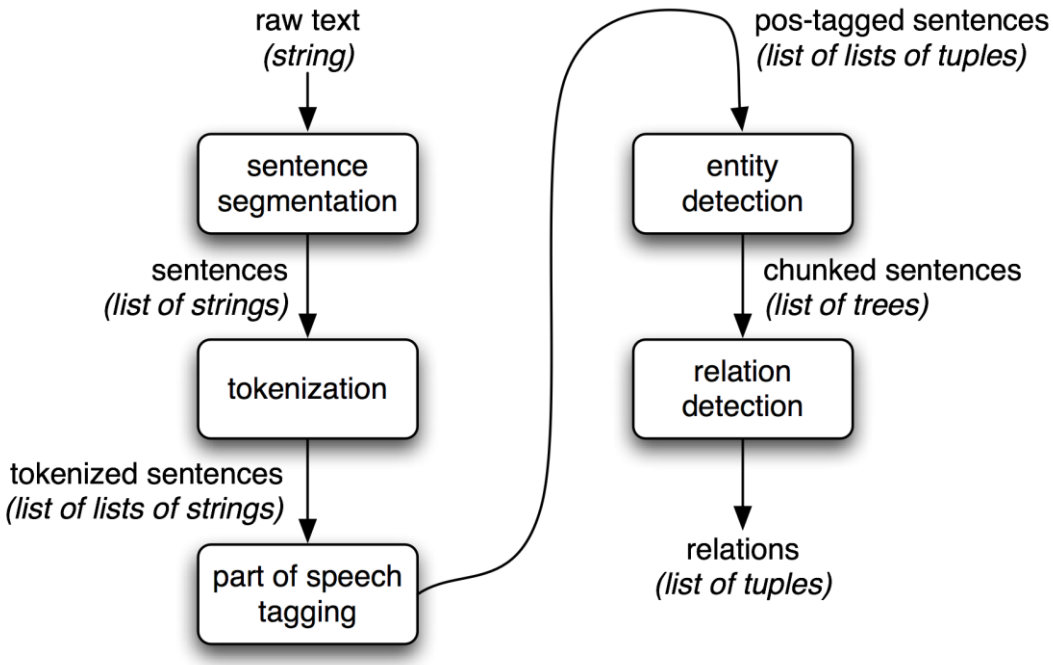


Figure 2.8: Pipeline for an information extraction system using NLTK (Bird et al., 2009)

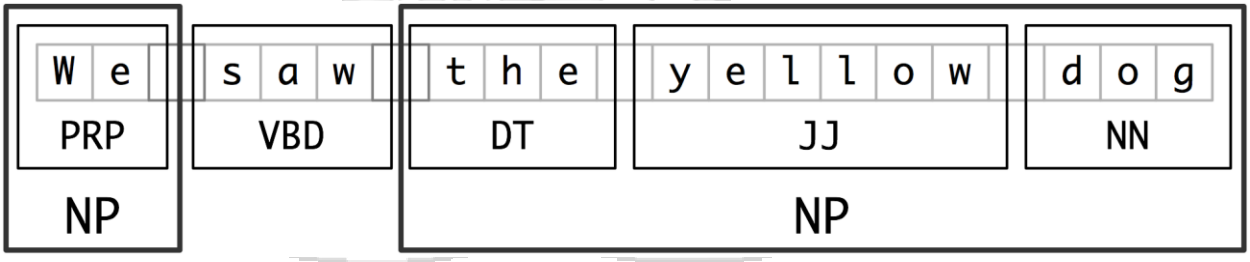


Figure 2.9: Segmentation and labeling at the token and chunk levels (Bird et al., 2009)

2.2.5. Event Extraction.

Event extraction involves the identification of events in free text and deriving detailed information about them; identifying who did what to whom, when, with what methods, where and why (Piskorski et al., 2008). The tools and techniques mentioned in previous sections of this chapter are essential in the identification of events reported in free text. Such event identification and extraction of detailed information about these events is important in order to create structured data regarding events reported in newswire content. Event extraction is a higher-level information extraction task which is complicated due to the fact that in news a full event description is usually scattered over several sentences and articles (Piskorski et al., 2008).

Additionally, nested event structures are a common occurrence in both open domain and domain specific extraction tasks; for example, a crime event can cause an investigation event which can lead to an arrest event (McClosky, Surdeanu & Manning, 2011).

Accuracy from event extraction is affected by the fact that a single event may be reported across a number of newswire articles. This poses a risk of repetition of details leading to inaccurate data. For example, if a car accident is reported on a Monday and then on Wednesday a new article is posted reporting on the condition of the victims who were hospitalized, the event extraction system might detect an accident event in the second article but not link it to the first article. This may lead to a new event distinct from the first with its own victim count. In some systems since information about events is scattered over different articles, single pieces of information are validated and aggregated at a cluster-level (Piskorski et al., 2008).

2.3. Existing Information Extraction Systems

2.3.1. Python Engine for Text Resolution and Related Coding Hierarchy.

2.3.1.1. Overview of PETRARCH.

Python Engine for Text Resolution and Related Coding Hierarchy (PETRARCH) is a descendant of the KEDS and TABARI systems. KEDS was used to generate a 12-year event data series for the Middle East generating a data set of about 60,000 events (Schrodt, Davis, & Weddle, 1994). This section will describe some of the features of PETRARCH and it strengths and weaknesses from the point of view of event extraction and information utilization. PETRARCH is a natural language processing tool for machine-coding events data (Schrodt, Beieler, & Idris, 2014). Relations of who did what to whom are extracted from news summaries (Open Event Data Alliance, 2015). Event data are used to reduce journalistic descriptions to categorical data that can be analyzed statistically (Schrodt et al., 1994). PETRARCH collects raw unstructured text using a web scraper that automatically pulls news stories from RSS feeds and then extracts structured data from the unstructured texts using an event data coding system (Open Event Data Alliance, 2015). This process of structuring unstructured data makes it easy for computer interpretation which opens up to many more possibilities of how this data can be analyzed and presented.

2.3.1.2. Strengths of PETRARCH.

The upgrade from TABARI to PETRARCH included functionality improvement which enabled PETRARCH to extract noun phrases corresponding to actors not in the defined system dictionaries (Open Event Data Analysis, 2015). This is an important feature as it enables the system to learn and adapt to new event information, specifically on new actors that are involved in every day events. The source actors and target actors keep changing every day. Companies rebrand and new terrorist group emerge every few year. If an event extraction system depended only on already documented actors many events would be missed leading to serious inaccuracy in extracted data.

2.3.2. Virtual Research Associates Reader.

2.3.2.1. Overview of the VRA reader.

The reader developed by Virtual Research Associates (VRA) is a tool for processing data from selected news stories and attempts to deliver a compact quantitative summary of all the events that are described. It uses the first sentence in an article to produce the quantitative summary thus taking advantage of a common practice in journalism in which reporters write lead sentences that summarize the key points in the article. Lead summaries are represented by the reader as database records with the fields for source and target actors for each event. Events are coded into a category typology called the Integrated Data for Events Analysis (King & Lowe, 2003). IDEA framework includes all the event forms, actors, and targets of earlier events frameworks like WEIS and PANDA. It uses a four-level event hierarchy to include new event forms as specifications of more general event forms (Bond, Bond, Oh, Jenkins, & Taylor, 2003).

The VRA reader thus demonstrates how NLP can be used to automatically summarize events to enable quick utilization of large information sets for analytical purposes. The IDEA protocol and the VRA Knowledge Manager Software system operate together to generate social, economic, environmental, and political events data and to display them in summary form in terms of event counts and various scales (Bond et al., 2003). Summarizing these events in tables, graphs, and maps enables analysts to quickly gauge the trend of events in an ongoing situation. The VRA Knowledge Manager enables users to review in-depth reports of data-points on these tables, graphs, and maps. It therefore provides easy to observe overviews of trends as well as close grained analyses of specific event sequences (Bond et al., 2003).

2.3.2.2. Strengths of the VRA reader.

Custom datasets can be generated at will. A data on demand approach better facilitates the incorporation of ongoing improvements in measurement and offers data more appropriate to specific research questions (Bond et al., 2003). Bond et al. (2003), offer a comparison between human coding for information extraction and automated information extraction using the IDEA framework and NLP techniques; they state that human coders tend to ignore grammatical literalism in defining an event while machines do not infer implied events and do not miss events simply because they are entangled grammatically with other events. Protocol improvements are permanent and cumulative, thus as more research discovers improvements they are added into the existing system over time to improve it rather than create a whole new system and this works well with the desire preserve the system's backwards compatibility and extensibility (Bond et al., 2003).

2.3.3. Europe Media Monitor.

2.3.3.1. Overview of the EMM.

EMM is an application of a near real-time multilingual news monitoring and analysis system research and developed at the Joint Research Centre of the European Commission for extracting violent and natural disaster event information from internet news sources (Piskorski et al., 2008). It contains a number of different modules that operate together with the aim of extracting information from various news sources and presenting it (Atkinson & Van der Goot, 2009). Some of the processes that take place in the EMM system are; meta-data creation, real time clustering of news articles, geo-locating or geo-tagging of news articles, core linguistic analysis, and event type classification among others (Piskorski et al., 2008). This is demonstrated in Figure 2.8 which shows the event extraction processing chain.

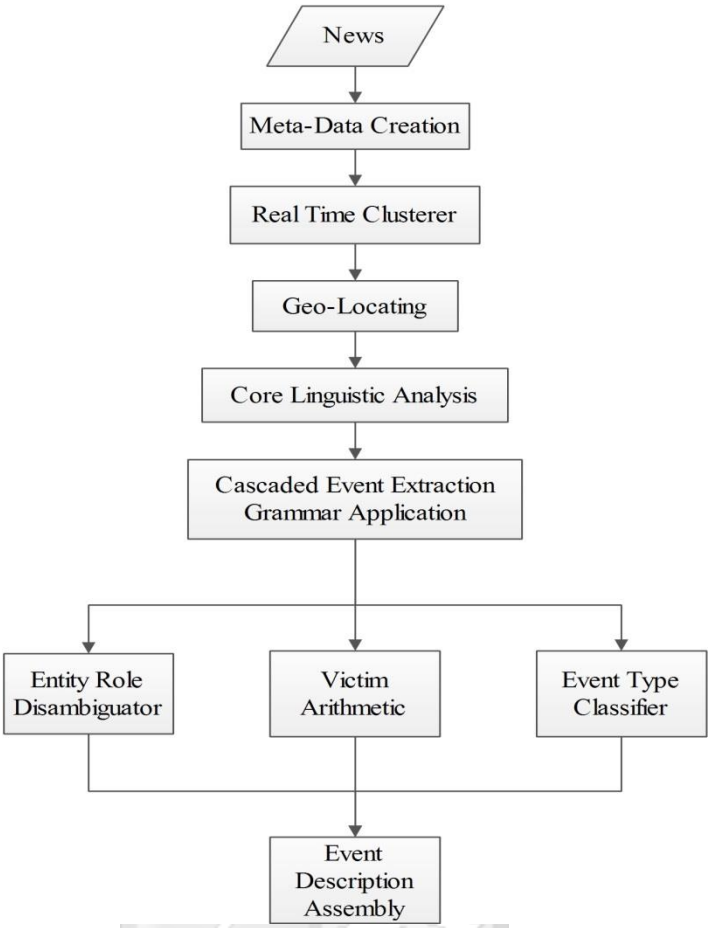


Figure 2.10: EMM real time event extraction processing chain (Piskorski et al., 2008)

2.3.3.2. Features of EMM.

This section will cover only the features of EMM relevant to this research either because the proposed prototype will borrow from them or will try and improve on them. The EMM uses a web-scraper to scan 800 websites up to every 15 minutes and processes the incoming news articles from these websites. The sites to monitor are configured into the system as well as a recipe that tell the web-scraper how to analyze each site (Best, Van der Goot, Blackler, Garcia, & Horby, 2005). Best et al. (2005) describe a 3-step process with which the EMM scraper operates:

i. Removing non-standard and unnecessary tags from the HTML documents.
ii. Converting HTML into XHTML.
iii. Transforming XHTML into RSS using a style sheet for each site.

Another interesting feature of EMM is the News tracker. News trackers follow a particular subject of a given event over an extended period of time. Thus one is able to analyze a subject after an event has occurred by analyzing past articles (Best et al., 2005). ExPRESS is a highly efficient extraction pattern engine used in EMM that is capable of matching thousands of patterns against megabyte-sized texts within seconds (Piskorski et al., 2008). An Express grammar consists of pattern-action rules; where the left hand side of a rule also known as the recognition part is a regular expression over flat feature structures and the right hand side of a rule also known as the action part constitutes a list of flat feature structures which will be returned in case the left hand side pattern is matched (Piskorski et al., 2008).

2.3.3.3. Strengths of EMM.

The news on EMM is not transient, meaning articles are stored and are always accessible which avoid the problem of broken links (Best et al., 2005). Broken links are a problem especially when someone wants to refer to the original document from which a certain event was extracted from.

2.4. Challenges Faced by Existing Information Extraction Systems

The PETRARCH event coder has been reported to be slower than its predecessor TABARI. It has a Stanford CoreNLP processing stage that poses a bottleneck due to a problem parallelizing Stanford CoreNLP and the need to interoperate the Java code of Stanford CoreNLP and the Python codebase of PETRARCH (Schrodt et al., 2014). Automated coding entails the hazard of duplication whereby, if the same event is reported in multiple stories the machine will generate multiple event records (Bond et al., 2003). The risk is greatest with crisis events, such as coup d'état, or a protracted process, such as a national election, that generate repeated references to the same real world events or processes, often filed by news reporters on the same or subsequent days (Bond et al., 2003). A common problem with machine event extraction tools is that nuance and context specificity are lost (Bond et al., 2003). This means some accuracy may be lost while reporting certain types of events that are highly context dependent.

2.5. Conceptual Framework

Not all the existing NLP systems were covered as there was a finite amount of time to conduct this research and a multitude of systems that have been developed. This research instead focused on the main systems that have been developed and upon which other systems are based. Figure 2.11 represents a conceptual framework of the processes and techniques that were the focus of this research.

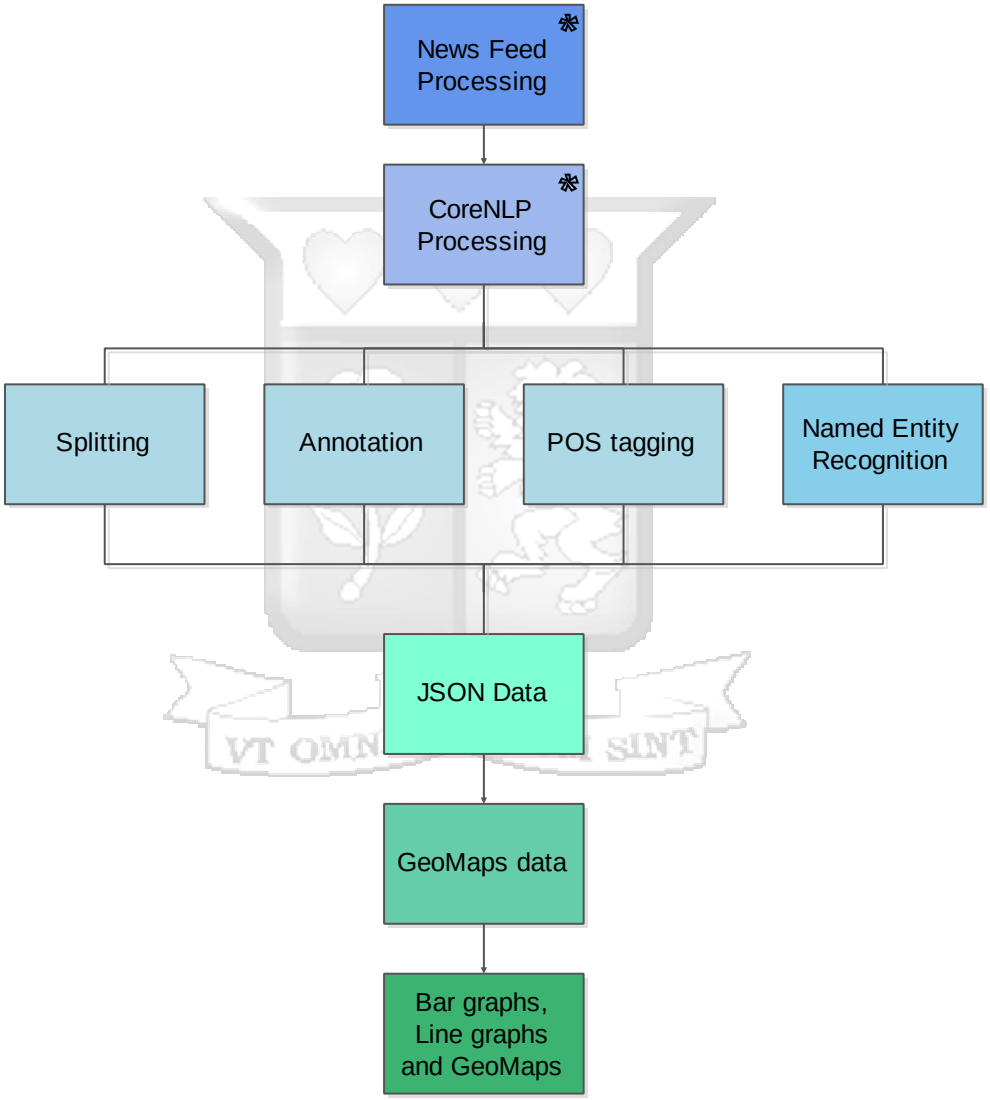


Figure 2.11: Conceptual framework

Figure 2.11 shows a conceptual framework of the prototype that would eventually be developed. The first stage would be processing of the raw news feed which would be acquired

from RSS feeds. This processing would involve XML and HTML parsing to extract plain text from articles. In the second stage the extracted text would go through processing using the Stanford CoreNLP processor which would include splitting, annotation, part-of-speech tagging, and named entity recognition. The information extracted would then be formatted into JavaScript object notation (JSON) for easy interpretation by the graphical user interface which would include geographical maps, bar graphs, and line graphs.

VT OMNES
VNVM SINT

Chapter 3: Research Methodology

3.1. Research Design

This was an applied research that tried to find a solution to the problem of inadequate methods of utilizing news media content over a long period of time. This research was conducted in stages that covered specific aims and objectives. These stages were recursive to ensure that all aspects of the research were well covered. The first stage in the research was to focus the study and to set the stage for the study (Alber, 2010). This included; selecting and narrowing down on a topic, identifying the problem statement, developing research questions and objectives, justifying the research, describing the context of the research , and identifying central issues surrounding the specific focus of the research project (Alber, 2010).

The second stage of the research was to write a review of the related literature (Alber, 2010). The second stage of the research took a well-rounded approach and covered various factors relating to the tools and techniques that were identified. This included the shortcomings of these tools and techniques, the level of advancement and the currently ongoing research involving these tools and techniques. A critical element of this stage was to identify the gaps that exist and try to address them. The second stage served three main purposes; to demonstrate an understanding of the field of study, to inform and shape the research project, and finally to prepare future readers to better understand the intellectual context of the study (Alber, 2010).

The third stage was creating an implementation plan that led to the generation of data and the development of a prototype that addressed the research objectives and answered the research questions (Alber, 2010). This stage described ways in which the research was to be conducted and included the following components; the setting of the study, the data-generating procedures and activities, the data to be collected to address the research objectives and answer the research questions, and a timeline for activities, data collection and prototype development (Alber, 2010).

The fourth stage was developing the prototype. At this point, the approach for the development of the prototype had been identified subject to the findings in the second stage of the research. The various system analysis and development techniques were adhered to so as to ensure a well-designed prototype that addressed the gaps identified in the second stage.

The fifth stage was testing of the prototype and analysis of data gathered during the third stage and while testing the prototype. This stage included; organizing, sorting, comparing, contrasting and categorizing the data collected and creating displays for the findings in clear and honest ways (Alber, 2010). Finally, the findings of the research were discussed as they related to the findings of other researches (Alber, 2010).

Figure 3.1: An overview of the steps involved in the research execution

3.2. Research Setting

3.2.1. Geographical Location.

The research was conducted in Kenya. The newswire content to be analyzed came from news reported from all regions in Kenya. The research focused on Nation Media Group digital content, specifically the Daily Nation website. The articles that were focused on were those that reported local events that occurred within Kenya. Data collection were also be narrowed down to Nation Media Group. Any data collection methods and tools were applied within the organization with permission from the relevant parties.

3.2.2. Mission and Philosophy of the Organization.

The mission of the organization is to create value for stakeholders and to positively influence society by providing media that informs, educates and entertains while the vision is to be the media of Africa for Africa. The mission and vision were relevant to this research because the benefits of the research could then have been aligned to the mission and vision of the organization.

3.2.3. Timeframe of the Study.

The research was undertaken over a period of 9 months; from June, 2015 to February, 2016. The newswire content focused on ran during those 9 months. That is, the content analyzed was content from June, 2015 to February, 2016.

3.2.4. Focus of the Study.

Due to the limited amount of time that was set for this research, the research only focused on a narrow spectrum of events. These included the most common reported events over the nine months of the research. This ensured a substantive analysis is done on the events and relations between events were more conclusive. The prototype design however allowed for scaling to cover additional events over time.

3.2.5. Anticipated Issues.

A number of possible issues had been anticipated within the research setting of this study:

i. Legal issues regarding the use of content belonging to the organization – Binding contracts would have to be drafted regarding how much of the content belonging to the organization can be used and to what extent.
ii. Non-disclosure agreements signed by the researcher may prevent revealing of details on certain processes within the organization – these had to be adhered to so as to abide by legal and ethical conduct.
iii. Unwilling research participants – this may be dealt with by elucidating beforehand the purpose and significance of the research.

3.3. Data Collection

3.3.1. Data Collection Overview.

Data collection was done in two phases. The first phase was to collect data required for the design of the prototype. The second phase was to collect data in relation to the developed prototype once it was complete. Data collection involved secondary data from online sources and books as well as the newswire content that was consumed by the prototype. Pilot-tests for the data collection instruments and data analysis procedures were done to identify potential pitfalls that may be faced (Easterbrook, Singer, Storey, & Damian, 2008).

3.3.2. Data Collection Process.

Two categories of data were collected for the system design and implementation phase of the research. The first category of data was data that would be consumed by the main module of the prototype which would use the Stanford CoreNLP framework to process and analyze content. Collection of this data was done by getting a list of URLs from the RSS feed of the source of newswire content. This was done once every 24 hours for the duration of the research. The RSS feed would contain an average of 70 URLs each leading to a unique article published on the Daily Nation website. The second category of data was data that would be used for the event detection module of the prototype. The type of event that was focused on was road accidents. For this category data collection involved analyzing a large number of articles that had been published of a span of about one year. Random articles were then be picked and labeled according to whether the event of focus was being reported or not. Articles in which the event was being reported were labeled as *1* and those where the event was not being reported were labeled as *0*.

3.4. Prototype Analysis, Development, and Implementation.

3.4.1. Requirements Gathering.

Data was collected prior to designing the prototype. This data was critical in coming up with system requirements that will guide the design process of the prototype. The research problem statement, research objectives, and research questions were also used as guidelines towards the formulation of the system requirements.

3.4.2. Prototype Design.

A number of System Analysis and Design tools were crucial for the design and development of the prototype to ensure all requirements were met and to have benchmarks upon which implementations of the prototype into a full system could be based upon (Dennis, Wixom, & Roth, 2012). Time planning tools were especially beneficial in ensuring the development and implementation of the prototype was done within good time. Project Libre is an open source tool that aids in project management and creating time plans for various projects. The choice of free and open-source software was important in keeping the costs of the research low without compromising on the quality of output that was generated by these tools.

		Name	Duration	Start	Finish	Predecessors
1		Compile prototype require...	5 days	11/2/15 8:00 AM	11/6/15 5:00 PM	
2		Create use case diagrams	2 days	11/9/15 8:00 AM	11/10/15 5:00 PM	1
3		Create data flow diagrams	2 days	11/11/15 8:00 AM	11/12/15 5:00 PM	2
4		Create entity relationship ...	2 days	11/13/15 8:00 AM	11/16/15 5:00 PM	3
5		Create UML diagrams	2 days	11/13/15 8:00 AM	11/16/15 5:00 PM	3
6		Design user interface	3 days	11/17/15 8:00 AM	11/19/15 5:00 PM	5
7		Develop prototype	10 days	11/20/15 8:00 AM	12/3/15 5:00 PM	6
8		Test prototype	5 days	12/4/15 8:00 AM	12/10/15 5:00 PM	7

Figure 3.2: Image showing work breakdown structure created using Project Libre

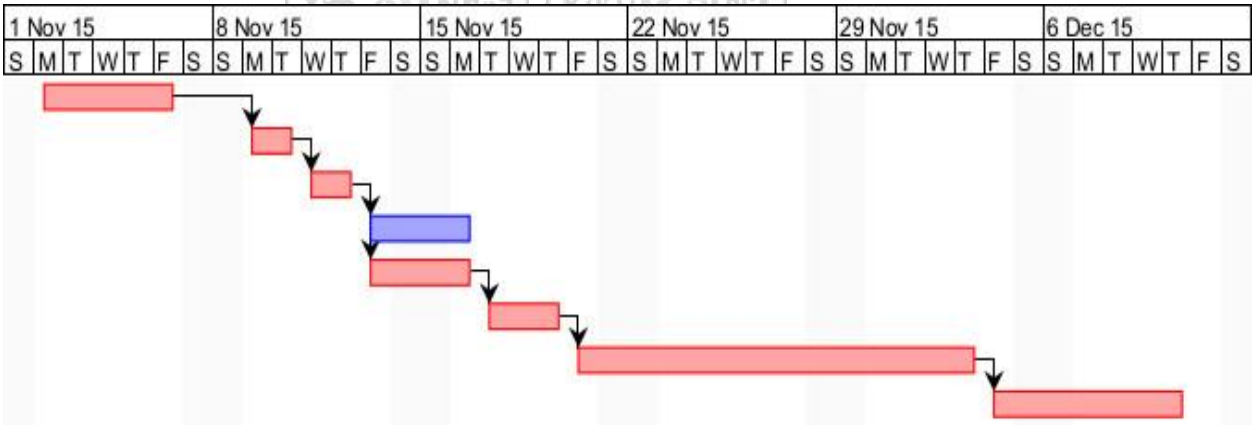


Figure 3.3: Gantt chart created using Project Libre

UML designer is free and open-source tool that is useful in designing and editing UML diagrams. It has provisions for use case diagrams, class diagrams, activity diagrams, and sequence diagrams among other useful features. Figure 3.4 shows a use case diagram with some of the use cases that will be included in the prototype.

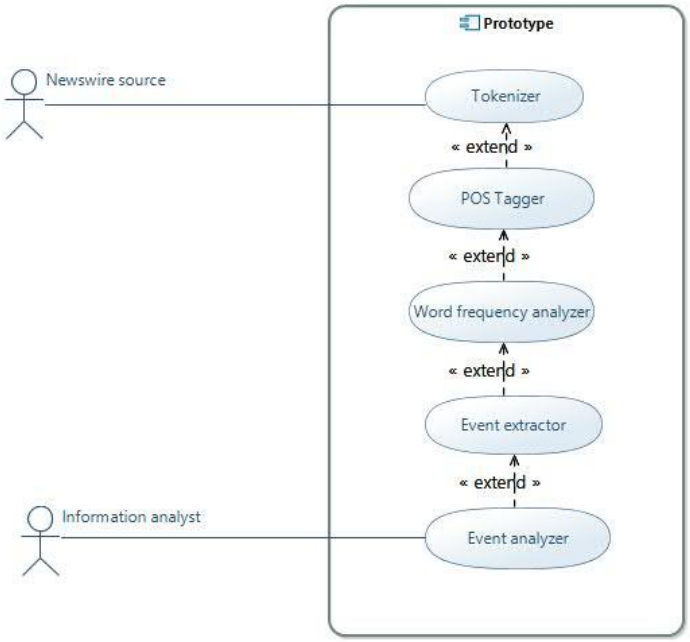


Figure 3.4: Use case diagram for the proposed prototype created using UML designer

3.4.3. Prototype Implementation.

From the design of the prototype, the most effective tools were identified for the development process. These included; APIs, Programming Languages, Operating Systems, Database Management Systems, and NLP tools that are most suitable to meet the design requirements of the prototype.

3.4.4. Prototype Testing.

A number of system testing methods were taken into consideration during the testing phase. The prototype was ideally meant to be interactive and hence graphical user interface testing and usability testing were essential. Software performance testing was also necessary

particularly in determining the scalability of the prototype and its feasibility as an implementable system.

3.5. Data Analysis and Presentation

Data Analysis was congruent to data collection. It was done in two stages. The first stage was analyzing the data gathered so as to formulate the requirements for the prototype design. The second stage was an analysis of the data generated during the testing of the developed prototype.

VT OMNES
VNVM SINT

Chapter 4: System Design and Architecture

4.1. Requirements Gathering

The requirements for the design and development of the prototype were formulated from the information that was gathered from the literature review as well as the problem statement. This information was instrumental in coming up with the requirements for the development of the prototype. The essence of the problem as identified in the problem statement was the ineffective utilization of newswire content over a long period time. In the literature review, existing systems that try to address the problem were identified and explored pointing out both their strengths and weaknesses. This step facilitated the identification of areas that the existing systems had not addressed. Additionally, techniques that could be used to address the problem were explored and this enabled identification of what was feasible and what techniques could be applied to address the problem.

The requirements that were identified were as follows:

i. To design and develop a prototype that is able to parse text from news articles that are obtained from a news feed.
ii. To design and develop a prototype that is able to tokenize the text that is obtained from news articles into 'bag of words' format with corresponding word counts.
iii. To design and develop a prototype that is able to tag words with suitable part-of-speech tags using suitable tools.
iv. To design and develop a prototype that is able to perform Name Entity Recognition on POS tagged words.
v. To design and develop a prototype that will display the frequency of name entities in an easy to interpret format.
vi. To design and develop a prototype that will map out the distribution of location name entities on an interactive map.
vii. To perform event detection on the data that is generated by the prototype.

4.2. System Design

4.2.1. Prototype Architecture.

The prototype architecture is as shown in Figure 4.1.

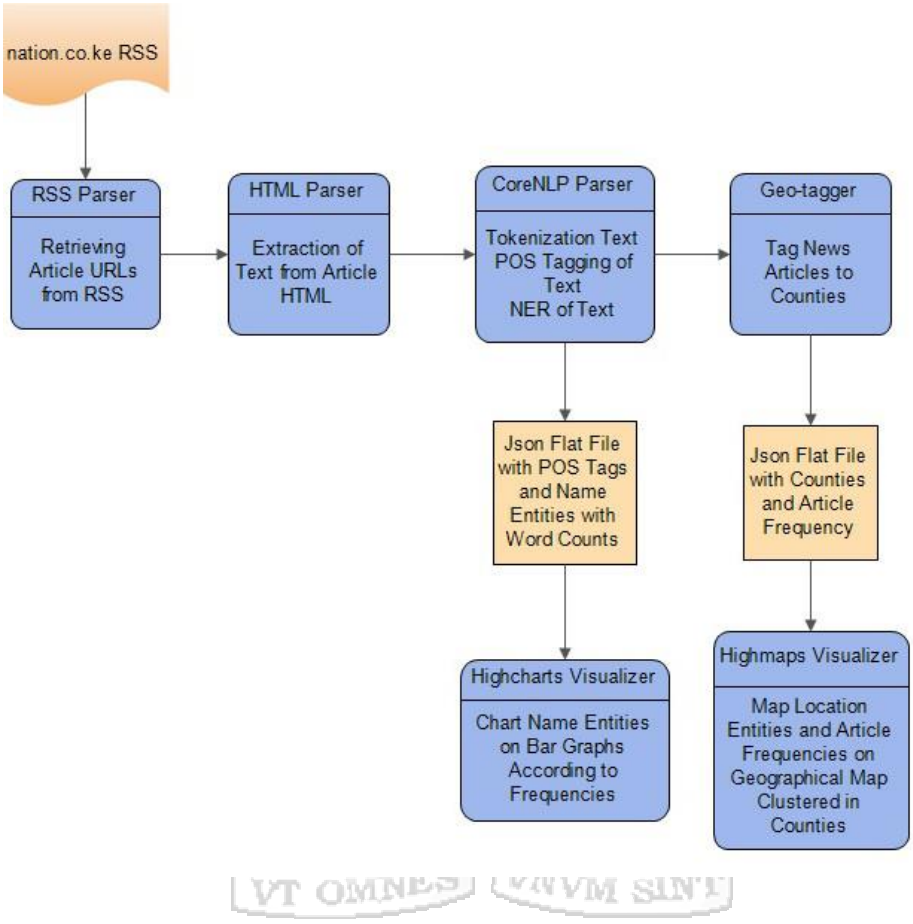


Figure 4.1: Data flow diagram showing the prototype architecture.

The prototype was designed to take input from an RSS feed which was sourced from the Daily Nation website with relevant permission from the content owners through a content user agreement shown in Appendix C. This input was in the form of an XML document which contained article titles, article summaries, and the URLs that linked to the news articles. Software design tools such as flow charts and class diagrams were crucial in the design stages of the prototype. These tools helped evaluate the process flow of the prototype to be designed, and enabled the streamlining of these processes to ensure faster development and implementation of

the final product. Some of the tools and technologies that were identified for the development of the prototype are as shown in Table 4.1.

Table 4.1: Technologies used for the development of the prototype.

Technologies	**Version**	**Purpose**
Java Development Kit	1.8	Java was identified as the most suitable language to perform the text parsing, json generation and Core NLP processes. This was mainly due to the fact that the CoreNLP package was written in Java and integration with Java was the most straight forward option.
Netbeans IDE	8.0.2	Netbeans was identified as the development environment of choice due to its relative ease of use and advanced features that enhanced the development process.
Stanford CoreNLP Java package	-	The CoreNLP package contains several libraries that perform various NLP functions. CoreNLP was identified as the package that was most suited to the specific goals of this research.
Jsoup Java library	1.8.3	Jsoup was used primarily to parse HTML documents to obtain cleaned article text and titles.
Gson Java library	2.3	Gson was used to format the derived data/information into JSON structures that were then written into flat JSON files.
JUnit java library	4.10	JUnit was identified as the most suitable test library to perform unit test on the modules of the prototype written in Java.
Python	3.4	Python was selected specifically to perform the tasks that could not effectively be performed

		using Java.
PyCharm IDE	5.0.1	PyCharm was identified as the most advanced alternative for a development environment for the purposes of this research.
Shapely Python package	1.5.13	Shapely was identified as the ideal package for geotagging and geolocation functions.
HTML	5	HTML was identified as the most suitable was in which to present extracted information visually. This was specifically due the relative ease of development and ubiquity of browsers that display HTML content.
JavaScript	-	JavaScript was used to perform operation on the JSON data that was generated by previous modules.
JQuery JavaScript library	1.11.3	JQuery simplified many of the functions that would have been relatively complex to perform using pure JavaScript and also provided additional functionality and features.
HighMaps JavaScript API	-	HighMaps provided features that enabled easy implementation and representation of data in interactive geographical maps.
HighCharts JavaScript API	-	HighCharts provided features that enabled easy implementation and representation of data in interactive graphs and charts.
Google maps Geocoding API	-	Geocoding API used location name entities captured by CoreNLP to provide longitude and latitude coordinates for locations.

4.2.2. Class Diagrams and Algorithms.

This research was designed more towards development of a prototype that was to be a proof of concept. For this reason, the number of classes was limited to greatly simplify the development and execution process. Figure 4.2 shows some of the classes and modules that were implemented in developing the prototype. The *Prototype* class was the main class and was written in Java programming language. This class included several methods as well as attributes. The attributes were many and therefore omitted in the illustration of the class diagrams for the sake of clarity. The main method of the *Prototype* class was the entry point during the running of the prototype application. This method performed several functions as outlined in the following algorithm;

1. *create a URL connection*
2. *process RSS feed contents – create list of article URLs*
3. **while** *list is not empty*
4. ***processUtterance****(list)*
5. *store the processed information*

Line one of the algorithm created a URL connection to the RSS feed URL and got an input stream for the contents found at the URL resource. This content was expected to be an RSS document with formatting similar to XML. Line two of the algorithm utilized an XML reader object to parse the RSS document and obtain the separate URLs for each of the articles that appeared in the RSS feed. Lines 3 and 4 called the *processUtterance* method which took a parameter that was a list of news article URLs to process. Line 5 stored the information returned from the *processUtterance* method into files that were labeled by date.

The *processUtterance* method was performed the majority of the NLP functions using the Stanford CoreNLP library module. The algorithm for the *processUtterance* method of the *Prototype* class was as follows;

1. *set properties for the CoreNLP parser*
2. **for each** *URL in the URL list*
3. *parse HTML document*
4. *tokenize the article text*

5. **for each** *token*
6. *get part-of-speech*
7. *get name entity*
8. **if** *token is location name entity*
9. *getCounty*
10. **return** *a JSON array with extracted information*

Line 1 of the algorithm set the properties that were required by the CoreNLP parser. Line 3 parsed the HTML documents found in each of the URLs passed to the *processUtterance* method and removed all the unnecessary tags and text to remain with a collection of the text, paragraphs, sentences, and words that made up the news article. Line 4 used the CoreNLP library to create tokens from the article text. In line 6 and 7, each of the tokens was then processed again by the CoreNLP library to obtain the part-of-speech tags and name entity tags for each token. Possible techniques for doing this were using an implementation of a Naïve Bayes algorithm or Hidden Markov Models having already been trained in the CoreNLP library. Line 8 checked if the token had been identified as a location. If it had, this location was mapped into one of the Kenyan counties using the *getCounty* method since the research was meant to focus on Kenya. Mapping the locations into counties would greatly improve the visualization of news distribution on a geographical map. The algorithm finally returned a JSON array containing all the extracted information in a well-organized format.

The algorithm to get the county of a particular location entity was as follows;

1. *get location coordinates*
2. *load GeoJSON file containing county sectors*
3. *get sector within which the location coordinates fall*
4. **return** *name of the county sector*

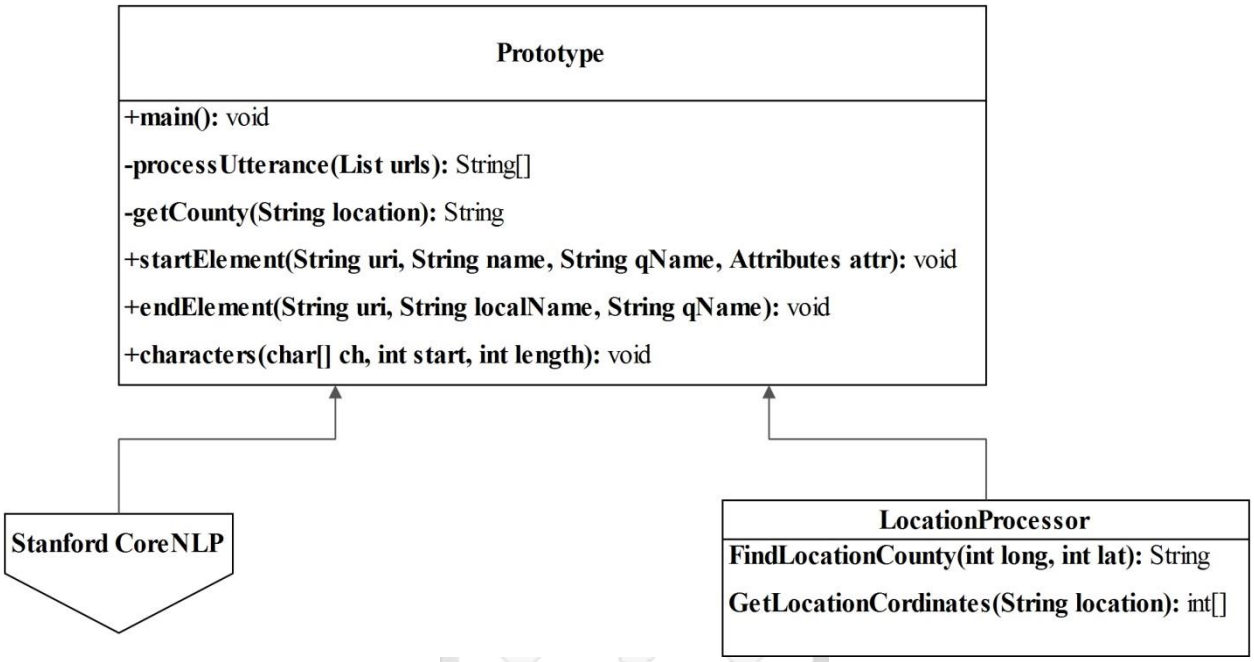


Figure 4.2: Class diagram showing the modules, classes, and various methods.

4.3. Process Flow

4.3.1. Overview.

The main framework used during the development of the prototype was the Stanford CoreNLP software. The Stanford CoreNLP software was selected because of its thorough documentation, ease of implementation, and robustness in terms of the techniques of NLP implemented within it. The relevant NLP tools required for the prototype that were available in Stanford CoreNLP were; part-of-speech tagger, name entity recognizer, parser, coreference resolution system, and sentiment analysis tools. Other tools and libraries were also used within the process flow of the diagram. Jsoup was used to parse HTML documents; shapely library for Python was used to map location information; while the Google Geomaps API was used to get location geographical coordinates. A complete table of these tools and libraries is available in Table 4.1.

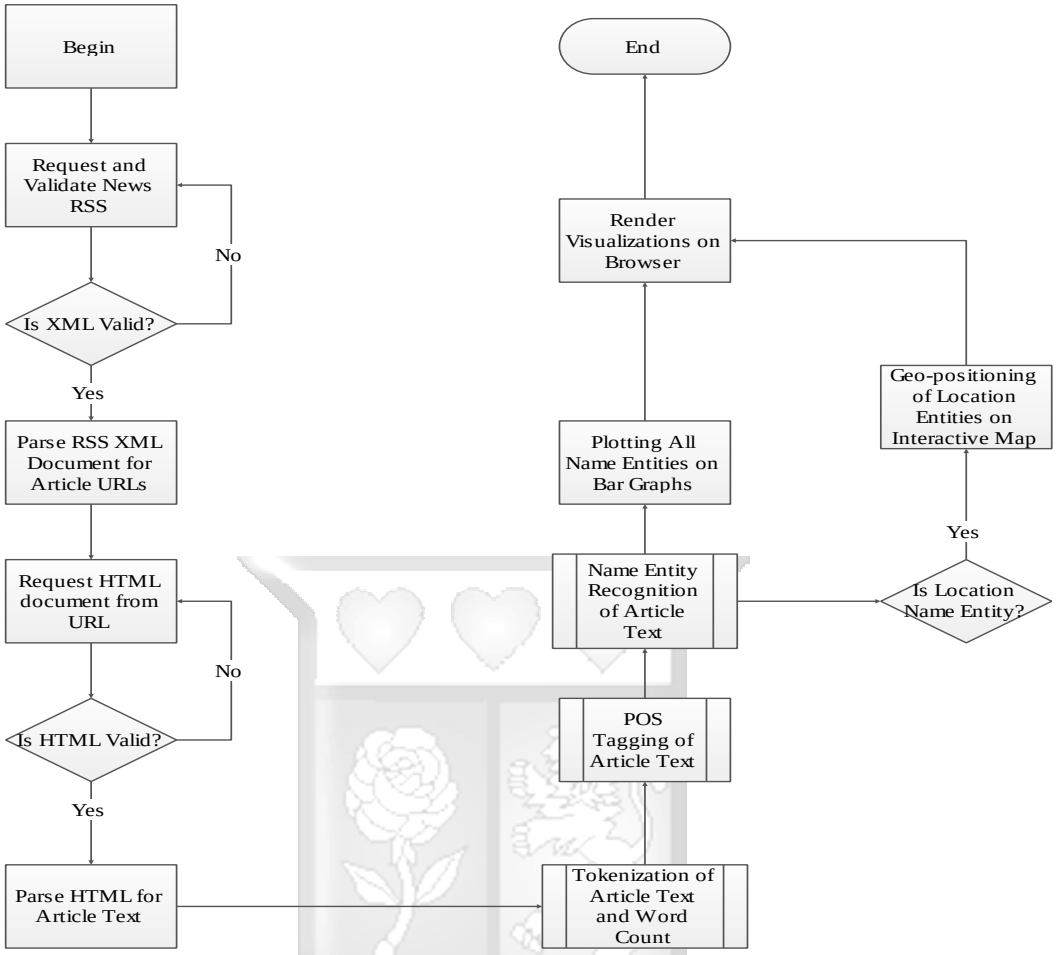


Figure 4.3: Prototype flow chart.

4.3.2. Text Parsing.

Text parsing involved extracting text from XML and HTML files that were fed to the prototype. The first step of this process involved extracting URLs from an RSS feed provided to the prototype. This RSS feed contained URLs of all of the day's news articles structured in an XML. This made it possible to extract the URLs and the title of the articles they led to by identifying the right XML tags in which they were contained. The XML parsing was handled by the Java Jsoup library.

```
<item rdf:about="http://www.nation.co.ke/sports/athletics/Athletes-urge-speedy-action-against-doping-to-avoid-ban/-/1100/3082612/-/2js8dp/-/index.html">
  <title>Athletes urge speedy action against doping to avoid ban</title>
  <link>http://www.nation.co.ke/sports/athletics/Athletes-urge-speedy-action-against-doping-to-avoid-ban/-/1100/3082612/-/2js8dp/-/index.html</link>
  <description>Athletes implore AK and Govt. to fast-track legislation criminalising doping to avoid Kenya ban.</description>
  <dc:date>2016-02-18T16:00:00Z</dc:date>
</item>
<item rdf:about="http://www.nation.co.ke/news/education/Pay-examination-fee-now/-/2643604/3083196/-/vf5miu/-/index.html">
  <title>Pay examination fee now, Knec tells private candidates</title>
  <link>http://www.nation.co.ke/news/education/Pay-examination-fee-now/-/2643604/3083196/-/vf5miu/-/index.html</link>
  <description>The government only pays examination fee for candidates in public schools.</description>
  <dc:date>2016-02-18T15:08:17Z</dc:date>
</item>
<item rdf:about="http://www.nation.co.ke/news/Officer-accuses-seniors-meddling-Embu-Speaker-kidnap-case/-/1056/3083188/-/3btqxiz/-/index.html">
  <title>Officer accuses seniors of meddling in Embu Speaker kidnap case</title>
  <link>http://www.nation.co.ke/news/Officer-accuses-seniors-meddling-Embu-Speaker-kidnap-case/-/1056/3083188/-/3btqxiz/-/index.html</link>
  <description>In letter to Chief Justice, man blames his legal woes on family feuds and business rivalries.</description>
  <dc:date>2016-02-18T15:03:29Z</dc:date>
</item>
```

Figure 4.4: RSS sample showing content items with standard XML tags.

The second step involved using each of the extracted URLs to obtain the HTML of the articles they led to. The article HTML pages contained a lot of information, a lot of which was not relevant to the focus of the research. Parsing of the HTML document was therefore necessary to extract only the article text which was to be analyzed by the prototype. The HTML parsing was handling was handled by the Java Gson library.

```
<article class="main column">
  <div class="story-view">
    <div class="section-home-header clearfix"></div>
    <header></header>
    <article class="article">
      <section class="article-sidebar"></section>
      <section class="body-copy">
        <section class="author"></section>
        <div itemprop="articleBody">
          <div></div>
          <div>
            <p class="MsoNormal">
            Junda MCA Paul Onje was arrested after Mr Shem Shefa, the organising secretary of Migori County Youth Forum, was killed by unknown
            people outside a pub in Rongo town after a quarrel.
            </p>
          </div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
          <div></div>
        </div>
      </section>
      <section></section>
    </article>
```

Figure 4.5: HTML sample showing tags that have to be parsed to retrieve article text.

4.3.3. Text Tokenization.

Text tokenization involved segmenting of the news articles into their constituent components. As explained in Chapter 2, text tokenization involves subdividing a text article into its component units; words, sentences and paragraphs. The Stanford CoreNLP tokenizer was able to segment the parsed articles into paragraphs, sentences, and words. Within the sentences the tokenizer was also able to identify n-grams which are word tuples that go together in a sentence. In such cases, words such as University of Nairobi would be identified as a single token. This was very important particularly to be able to perform part-of-speech tagging.

4.3.4. Part-of-Speech Tagging.

Part-of-speech tagging involved tagging or identifying words within the extracted articles according to their part of speech. This means that words would be given tags such as; noun, verb, adverb, adjective, and so on depending on where they appeared in a sentence and how they were used. Many words in the English language have multiple meanings and getting the intended meaning of a word as used in a sentence may be easy to a human being but can be challenging to a machine. For this reason techniques such as Hidden Markov Models are instrumental in determining the intended meaning of a word used in a sentence and consequently its part-of-speech tag.

Figure 4.6: Screen capture of JSON document showing POS tagging and word count of different words.

4.3.5. Name Entity Recognition.

Name entity recognition involved identifying name entities in free text. These included organizations, people, locations, and dates.

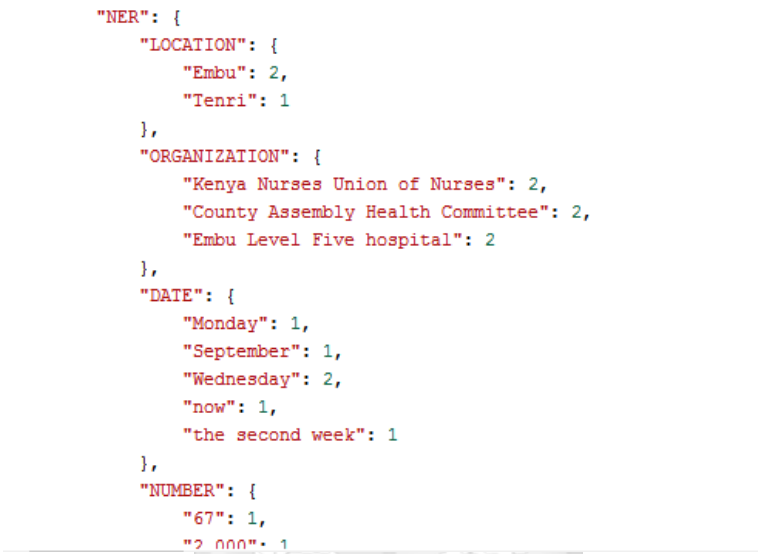


Figure 4.7: Screen capture showing NER for different name entities and their various counts.

4.3.6. Visualizations.

Visualizations comprised of bar graphs, and geographical maps. These enabled the lucid representation of extracted information. The representation of extracted information visually was important since the information needed to be easily interpreted to make sense to any user of the information without consuming too much time. The graph and map APIs that were used to implement the visualizations were HighCharts and HighMaps. These enabled implementation of clear and interactive graphs and maps that served the intended functional purpose of the research. The bar graphs showed the distribution and frequency of named entities and also gave a breakdown of the specific names and their frequency within each particular day. The geographical maps showed the distribution of news stories across the country by clustering mentioned location name entities into their respective counties. Placing a mouse pointer over a particular county would then give further details on the number of times various locations within the county were mention in news articles.

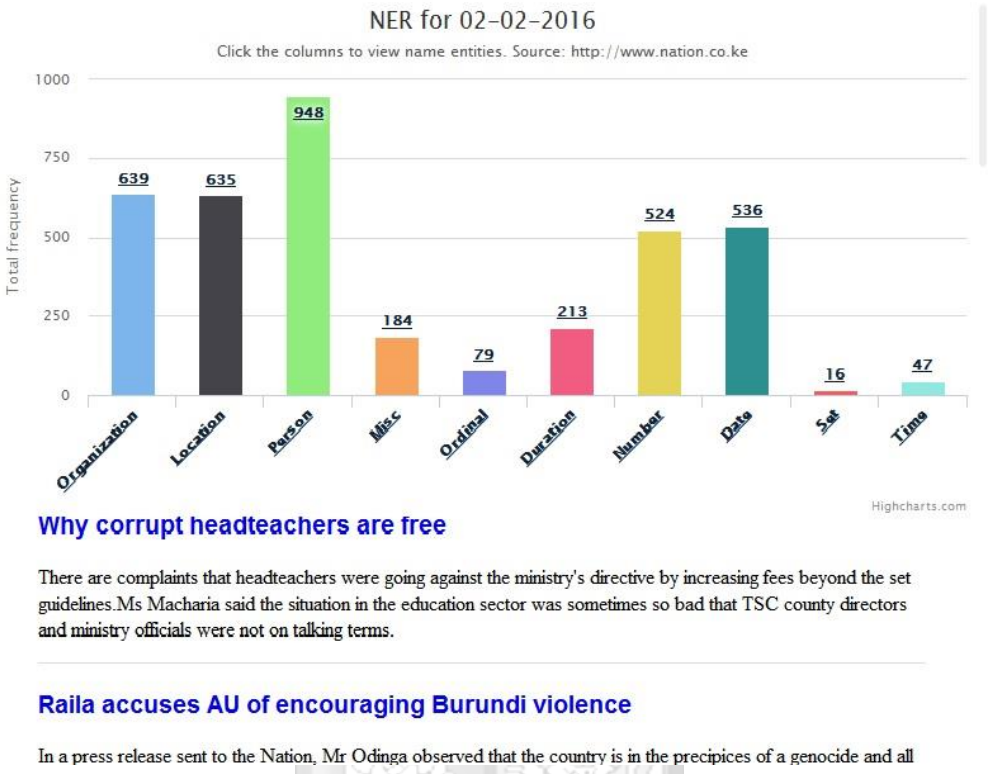


Figure 4.8: Screen capture showing bar graphs representing various name entities.

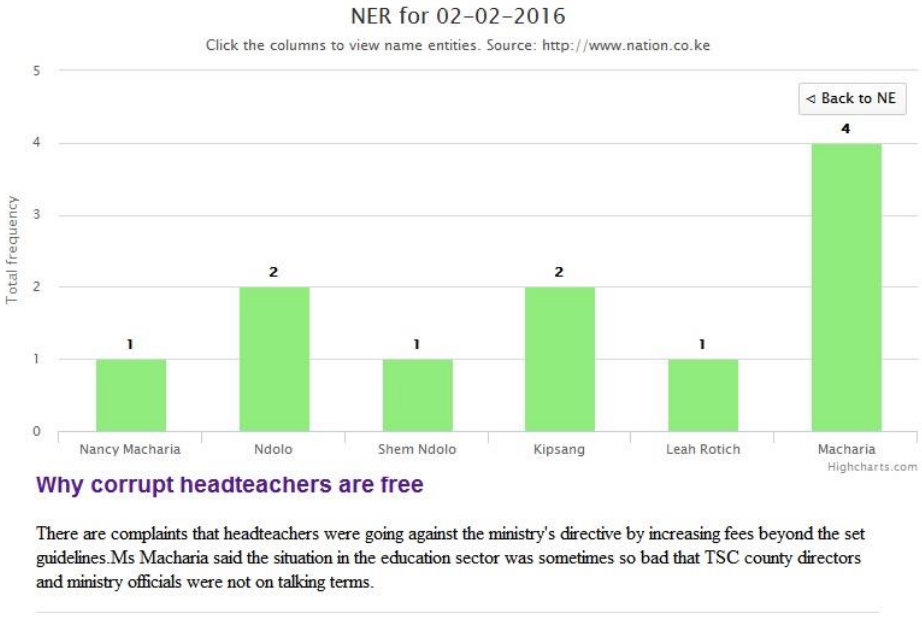


Figure 4.9: Screen capture showing bar graphs representing different members of a name entity.

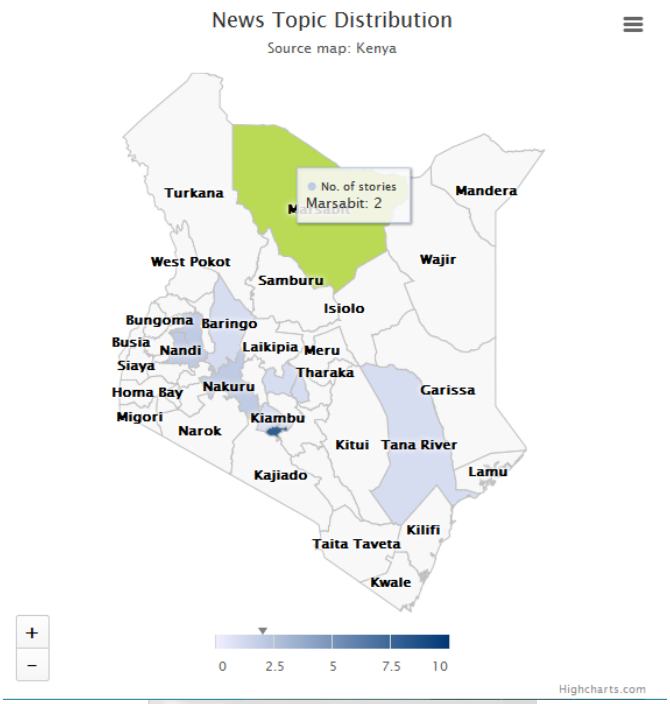


Figure 4.10: Geographic map showing news topic distribution across the area of focus.

4.4. Event Detection

4.4.1. Module Architecture.

The event detection module of the prototype was design independently of the other modules. This ensured that the development and implementation of this module would not delay the implementation of the rest of the modules due to the fact that the event detection module was more complicated and tasking to design and implement. Figure 4.11 shows the overall architecture of the module. The event detection trainer was designed to run a logistic regression algorithm over a training data set. The training data set was created by getting a list of article URLs. The articles were all read to determine if they were reporting on a particular event. Those that were reporting on an event were labeled as 1 those that were not were labeled as 0. The URLs were then processed to obtain data in a format that was more machine interpretable and took up less memory. The training algorithm was then run over the obtained data and prediction parameters were obtained. The parameters obtained were then applied by the event detector to detect events from articles that were input to it.

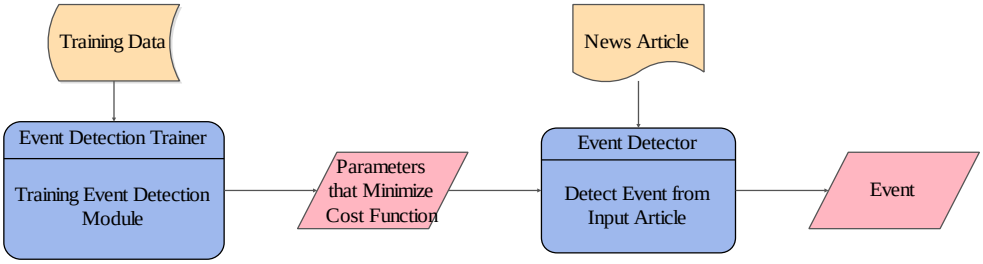


Figure 4.11: Architecture diagram for the event detection module.

4.4.2. Process Flow.

Figure 4.12 shows the process flow that was followed in training the module.

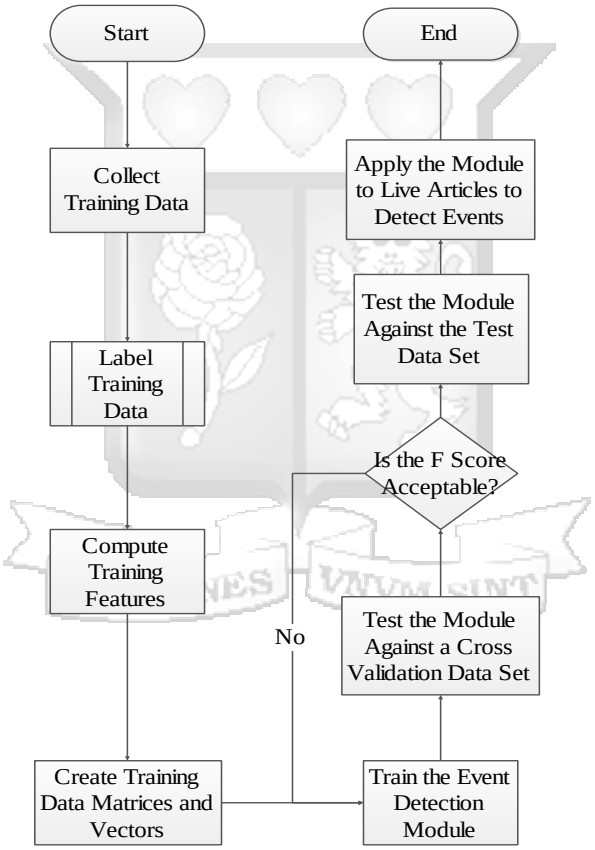


Figure 4.12: Flow chart showing the process flow for training the module.

4.4.3. Learning Model Design.

To create the event detection module, a logistic regression learning algorithm had to be implemented to train the module to detect particular events in articles. An initial data set of 105 articles was collected and labelled.

4.4.3.1. Logistic regression algorithm.

The hypothesis for logistic regression is defined as:

$$h_\theta(x) = g(\theta^T x) \tag{1}$$

Function g is the sigmoid function defined by:

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

The cost function for logistic regression is defined as:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right) - \left(1 - y^{(i)}\right)\log\left(1 - h_\theta\left(x^{(i)}\right)\right)\right] \tag{3}$$

And the gradient of the cost function is:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)} \tag{4}$$

The aim of the algorithm was to minimize the cost function and achieve high prediction accuracy. This was achieved by using the Broyden-Fletcher-Goldfarb-Shanno algorithm implemented in the scipy library *fmin_bfgs* optimization function to get the parameters that would give the lowest value when applied to the cost function. These parameters were then applied to input data sets to obtain predictions of events. Several iterations were made to improve the accuracy of the learning module. The collected data sets were divided in a ratio of 60:20:20. 60% was the training data set, 20% was the cross validation set, and the remaining 20% was the test data set.

The overall algorithm is as follows;

1. **do**
2. *load training data*
3. *compute cost*

4. *optimize cost function*
5. *load cross validation data*
6. *compute cross validation cost*
7. *compute accuracy*
8. *make data adjustments*
9. **while** accuracy < 85%
10. *load test data set*
11. *compute final accuracy*
12. *detect event in input articles*

The second line in the algorithm loaded the training data into an *m* by *n* matrix *x*, where *m* was the number of training examples while *n* was the number of features. The test output was also loaded into an *m* by 1 vector. The third line computed the cost using the cost function for logistic regression as well as the gradient using the logistic regression gradient function. The fourth line applied the *fmin_bfgs* minimization function to come up with the optimization parameters that were to be used in the prediction function. Lines 5 to 8 used the optimization parameters on a cross validation set and predicted the accuracy of the parameters. If the accuracy was below a selected threshold of 85%, necessary adjustments were made and the process was repeated until the threshold accuracy was achieved.

4.4.3.2. Learning module design.

From the module architecture and the process flow diagram, a suitable implementation design was settled on which would systematically perform all the required functions in a minimal amount of time and using minimal resources. The possible number of unique words was expected to increase almost linearly with the number of training examples. A training set with 85 training examples had about 5,000 unique words. For this reason the design needed to be optimized to minimize the running time of the module. Using ordinary list objects or linked lists without optimization to store and manipulate the words or tokens being worked on would have resulted in an order *n*, O(*n*), running time. With this in mind, if computation on each *n* item was to take 1 second then the overall running time on the 85 training examples would be about 5000 seconds. However applying a randomly built binary search tree with a worst case running time of O(*log(n)*) was able to reduce the overall predicted running time of the module. Despite the worst

case running time of the of a non-optimized linear binary search tree being O(*n*), the order of words that were in the training set were random enough to result in running time being closer to O(*log(n)*). From tests that were run, a red-black tree with a worst case time of O(*log(n)*) would have performed minimally better than the non-optimized binary search tree. The algorithm for inserting elements into the binary search tree was as follows (Cormen, Leiserson, Rivest, & Stein, 2009);

TREE-INSERT (Tree, z)

1. *y* ← *nil*
2. *x* ← *root[Tree]*
3. **while** *x is not nil*
4. **do** *y* ← *x*
5. *if key[z] < key[x]*
6. *then x* ← *left[x]*
7. *else x* ← *right[x]*
8. *parent[z]* ← *y*
9. **if** *y is nil*
10. **then** *root[Tree]* ← *z*
11. **else if** *key[z] < key[y]*
12. **then** *left[y]* ← *z*
13. **else** *right[y]* ← *z*

Left and right refer to the left child and right child of the current node. Parent refers to the parent of the current node while root refers to the root of the node or the whole tree. The insert function of the binary search tree algorithm runs in *O(h)* time where *h* is the height of the tree. Figure 4.13 shows the structure of a binary search tree with the keys 2, 3, 4, 6, 7, 9, 13, 15, 17, 18, and 20 inserted. The tree has a height of 5 and 11 nodes in total with 15 being the root node.

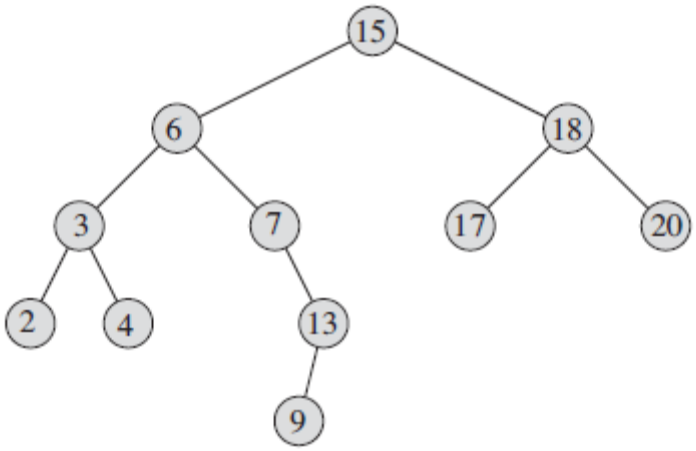


Figure 4.13: Binary search tree structure (Cormen et al., 2009).

The search algorithm for the binary search tree is as follows (Cormen et al., 2009);

TREE-SEARCH (x, k)

1. **if** *x is nil or k equals x.key*
2. **return** *x*
3. **if** *k < x.key*
4. **return** *TREE-SEARCH (x.left, k)*
5. **else return** *TREE-SEARCH (x.right, k)*

A *replacers* module created using functionality available in the NLTK library provided functionality to manipulate the text that was parsed from HTML articles. All this was performed in the *WordProcessor* class which provided the functions to parse the articles and load words into binary search trees that were then used by the *MatrixProcessor* to convert the text data into numbers which were easier to store and took up less space. This was done by generating a file that contained all words that appeared in the whole training set in order of the frequency of use. An array was then generated with the top n words. Each training set $x^{(i)}$ was the replaced by the index of each of the words from the array of most frequent words. This data was then written into a file that was saved to be used by the *ModuleTrainer* class.

Figure 4.14 shows the class diagrams of the learning module.

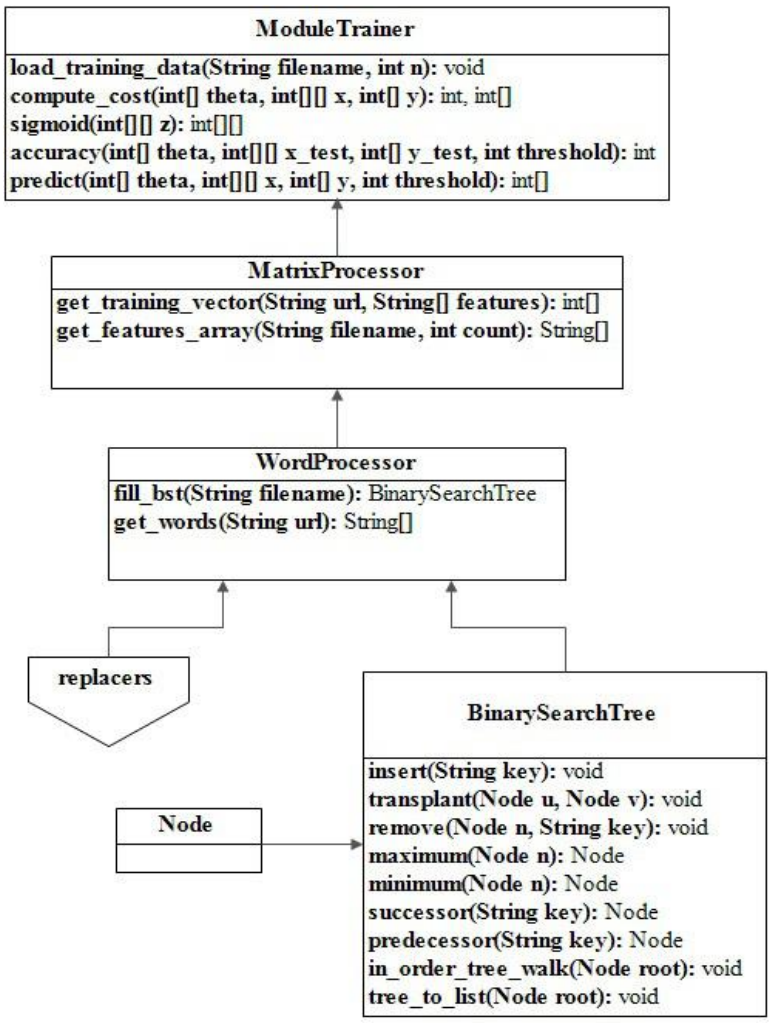


Figure 4.14: Class diagram for the event detection learning module.

VT OMNES
VNVM SINT

Chapter 5: System Implementation and Testing

5.1. Implementation

The main module of the prototype was programmed in Java. This was because the Stanford CoreNLP package was available mainly in Java libraries. This made the implementation and debugging easier and faster. The event detection module was implemented in Python primarily because of the availability of advanced scientific computation libraries such as scipy and numpy. The user interface portion of the prototype was done in HTML by integrating JavaScript as well as a variety of JavaScript libraries such as JQuery. HighCharts and HighMaps API were both an integral part of the user interface implementation that allowed visualization of large data sets as well as user interactivity that enhanced user experience. All source code was uploaded to the GitHub website both as a backup measure and to make it available to other researchers who would want to extend or modify it for their own purposes

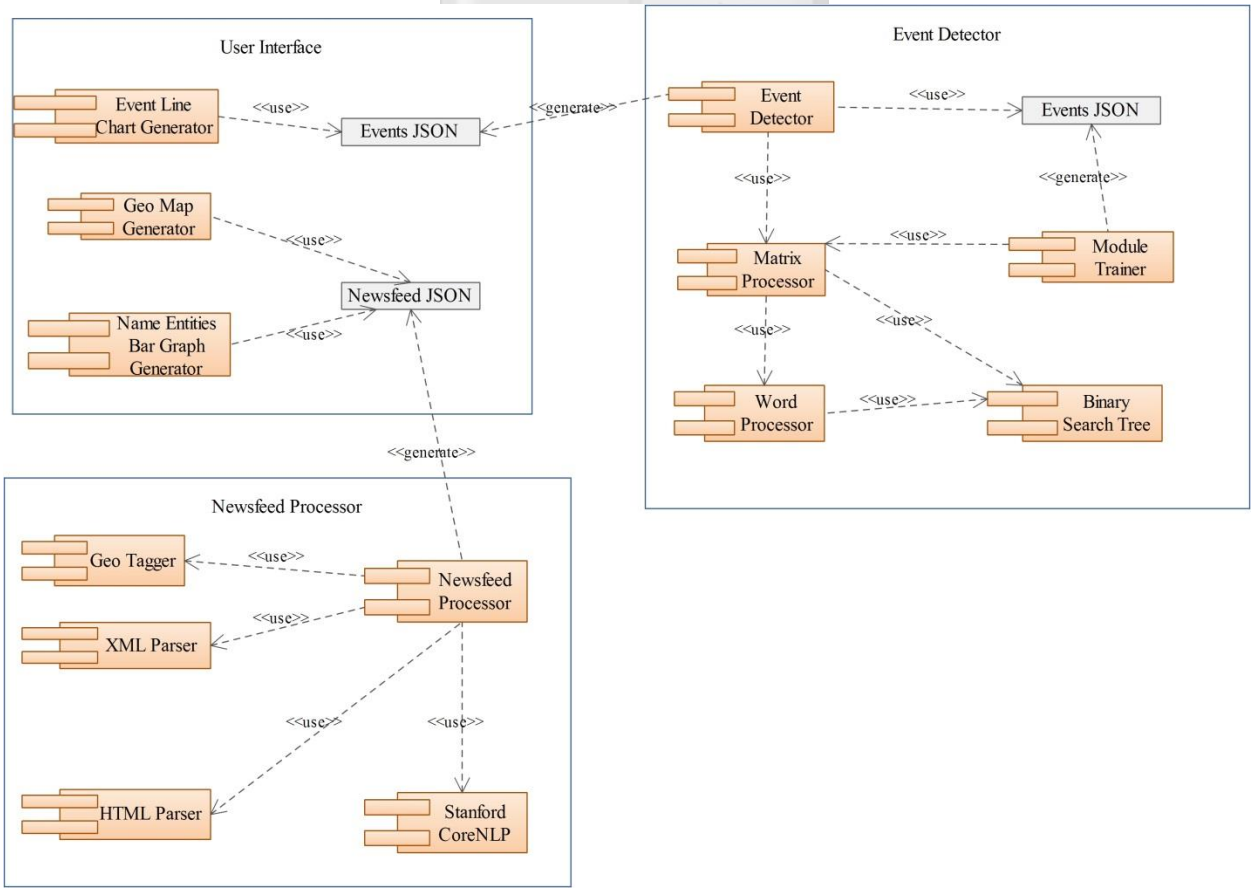


Figure 5.1: Implementation setup diagram.

Table 5.1: Project code bases and their associated links on GitHub.

Code Base	**GitHub URL**
User interface	https://github.com/smbuthia/news-analysis-prototype-dashboard
Event detector	https://github.com/smbuthia/logistic-regression-event-detector
Newsfeed processor	https://github.com/smbuthia/news-analysis-prototype-newsfeed-processor

5.2. Testing

5.2.1. Test Strategy.

The objective of performing tests was to ensure that the system requirements were met and that the prototype's modules and functionalities performed as per the requirements. Various rectifications and improvements were made during the testing process to improve on the overall performance of the prototype as a whole. The results of the tests were documented and tabulated in an easy to interpret manner. Any improvements that were made during the testing phase were also reported.

5.2.2. Main Module Testing.

5.2.2.1. Unit testing.

The main module was written mainly in Java programming language while the user interface and visualization components of the module were implemented in HTML and JavaScript. Due to the object oriented nature of Java programming language, unit tests were ideal for this module. The Java unit tests were implemented using the JUnit unit testing framework that enabled quick and simplified implementation of unit tests on existing classes. Table **5.2** shows the tabulated results of the unit tests that were performed on the executable functions.

Table 5.2: Tabulated results of Junit tests on Prototype class.

Method	**Warnings**	**Errors**	**Result**
getCounty	None	None	The method ran as it was supposed to and gave the correct county for each location that was passed to it as long as it was within Kenya.
processUtterance	None	None	The method ran as required and resulted in a correctly constructed JSON strings.
main	None	None	The method correctly called the other methods and stored the results in appropriate files.

5.2.2.2. Functionality testing.

Table 5.3: Tabulated results for functionality tests performed on the prototype.

Module	**Warnings**	**Errors**	**Result and Rectification**
location_processor	None	None	The module ran as expected and gave the correct location data. No rectification necessary.
word_processor	None	None	The module ran as expected and gave the expected output in form of a text file. No rectification necessary.
matrix_processor	None	None	The module ran as expected and generated the correct matrices and vectors for the text inputs given. No rectification necessary.
module_trainer	Possible divide by zero	None	The module ran as expected and produced a reasonable result with an initial accuracy of 85%. The divide by zero warning was as a result of getting the logarithm of zero which is undefined. Thus was rectified by setting a minimum value of 0. 00000001 while calling the logarithm function.

5.2.3. Event Detection Module Testing.

5.2.3.1. Performance testing – computation time.

The event detection module pipeline was as shown in Figure **5.2**

Figure 5.2: Event detection module pipeline.

The performance of each section of the pipeline was measured according to some predetermined parameters. The text processing segment included the process through which the text from articles in the training data set, cross validation data set, and test sets were converted into tokens and these tokens were subsequently represented numerically in vectors and matrices according to their frequencies. One shortcoming at this stage was the computational time, that is, the time it took for the system to process all the articles into the desired matrix that was suitable to be used as an input in the learning algorithm. One of the major causes of long computational times at this point was the number of words that had to be processed for each article. To try and address this issue, variations of the binary search tree were used. Initially, an unmodified search tree was used. This had a worst case running time of O(*log(n)*) since the words in the articles appeared in random order and thus there was little to no chance on the search tree resulting in a set of linearly connected nodes. For such a case the order of growth of the binary search tree was very close to that of the red-black tree, that is, O(*log(n)*). Table **5.4** shows a comparison of running times for unmodified BST and red-black tree. For this reason, the binary search tree was used in the final implementation due to the ease of implementation and debugging without compromising on the overall running time.

Table 5.4: Running times of module using different search trees.

Type	**Order of Growth (Worst Case Running Time)**	**Actual Running Time in Learning Module (ms)**
Binary Search Tree (Randomly Built)	O(*log(n)*)	1358.7430
Red-Black Tree	O(*log(n)*)	1248.6330

5.2.3.2. Performance testing – prediction accuracy.

The prediction accuracy of the learning algorithm was attributed to several factors. Among them were; the number of features used in the training set, the number of training examples, and the regularization parameter, λ. While other factors that could also have affected the accuracy of the learning module, these three were focused on and adjustments made to see what effect they would have on the prediction accuracy of the module. Initially a data set of 105 articles was used. These articles were collected over by searching through the Daily Nation website for articles reporting on road accident events as well as getting random articles from the website's RSS feed. This ensured that the training data was not skewed thus preventing issues while reporting the accuracy of the training module. The data was divided into two segments. The first was the training set and the second was the test set that was used to measure the accuracy of the trainer. The training set contained 85 articles while the test set contained 20 articles. As described in the section 4.4, the articles were converted to vectors that represented the words in the articles according to frequency. Each feature of the training set represented a word, and the number in the feature column represented the frequency of that word. Table 5.5 shows how the number of features affected the final cost, number of iterations, and accuracy of the trainer with the training data set held at 85 training examples.

Table 5.5: Training module accuracy with respect to number of features, n, with m set to 85.

Number of features (*n*)	**Cost at optimum**	**Iterations to get to optimum**	**Accuracy (%)**
10	1.33375E-05	80	75.00
15	5.87769E-06	25	65.00
20	5.12467E-06	27	70.00
25	7.09533E-06	25	70.00
30	1.00166E-05	21	75.00
35	2.48303E-06	21	75.00
40	1.04079E-05	19	85.00
45	5.77731E-06	20	85.00
50	9.70586E-06	19	85.00
55	2.64434E-06	20	85.00
60	2.65515E-06	20	85.00
65	2.71787E-06	20	85.00
70	2.78943E-06	20	90.00
75	2.56198E-06	20	85.00
80	2.44046E-06	20	85.00
85	1.22623E-06	21	85.00
90	2.31584E-06	20	85.00
95	2.25250E-06	20	90.00
100	2.02710E-06	20	90.00

At 70 features the reported cost was $2.78943 \text{x} 10^{-6}$ and it took 20 iterations of the minimization function to get the parameters that gave the optimum cost with a prediction accuracy of 90%. This figure was chosen as the ideal number of features and thus the number of features, *n*, was fixed at 70 and then 100. The next adjustment to be made was the number of training examples. Table 5.6 and Table 5.7 show the cost, number of iterations, and accuracy while varying the number of training examples.

Table 5.6: Training module accuracy with respect to number of training examples, m, with n set to 70.

Training examples (*m*)	**Cost at optimum**	**Iterations to get optimum**	**Accuracy (%)**
85	2.78982E-06	20	85.71
105	4.45439E-06	19	94.29
125	5.85526E-06	21	92.68
145	1.97386E-06	24	91.67

Table 5.7: Training module accuracy with respect to number of training examples, m, with n set to 100.

Training Examples (*m*)	**Cost at optimum**	**Iterations to get optimum**	**Accuracy (%)**
85	2.02730E-06	20	85.71
105	5.75774E-06	18	94.29
125	5.52822E-06	20	95.12
145	1.03734E-05	19	89.58

The accuracy of the learning module was generally expected to rise with an increase in training modules. The results of Table 5.6 and Table 5.7 were further analyzed in chapter 6. Finally, a regularization parameter λ was applied to the learning algorithm and varied. Table 5.8 shows the results of varying the regularization parameter. The cost and gradient formulae were adjusted to account for regularization as shown in equation 5 and 6.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right) - \left(1-y^{(i)}\right)\log\left(1-h_\theta\left(x^{(i)}\right)\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}{\theta_j}^2 \quad (5)$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\theta_j \quad (6)$$

Table 5.8: Training module accuracy with respect to a regularization parameter, with m set to 125 and n set to 100.

λ	**Cost at optimum**	**Iterations to get optimum**	**Accuracy (%)**
0.001	0.16786	3	92.68
0.01	0.16930	3	92.68
0.1	0.18368	3	92.68
1	0.32744	3	92.68
10	0.69024	2	92.68

Chapter 6: Discussions

6.1. Challenges

Some of the challenges that were experienced during the design and implementation of the prototype were:

i. The prototype was not sophisticated enough to filter out repeated articles and articles that reported on the same events.
ii. The HTML and RSS parser that was developed was rigid meaning significant changes to the HTML and RSS structure of the source of the newswire content would mean an update of the text parsing module of the prototype.
iii. Despite the significant advancement of the Stanford CoreNLP framework, its accuracy was limited to the accuracy to which it was trained to at the time of implementing the prototype. This means that it did not add new words to its training set and would repeatedly get some name entities wrong without the possibility of correction.
iv. The event detection module was observed to perform better if given a large training set to work with for each particular event. This meant that creating a training set for a large number of events to be detected would take tens of thousands of labeled news articles which would be both time consuming and resource intensive.

6.2. Challenge Resolutions

The inability to filter repeated articles is one that can be fixed by designing a system that can keep track of all articles that have been processed. In the same way binary search trees were used in the prototype to keep track of words and their frequencies, they can be used to keep track of URLs by comparing them to those that have already been processed. This would eliminate repeated articles being processed by the system. A better design for the HTML and RSS parser would reduce the need for having to update the text parsing module every time there is a change in the structure of the HTML and RSS document of the newswire sources. This would also enable the system to be extended to more sources without designing custom parsers for each newswire source. Figure 6.1 shows a class diagram with an improved design that enhances both the HTML parsing and RSS parsing functions.

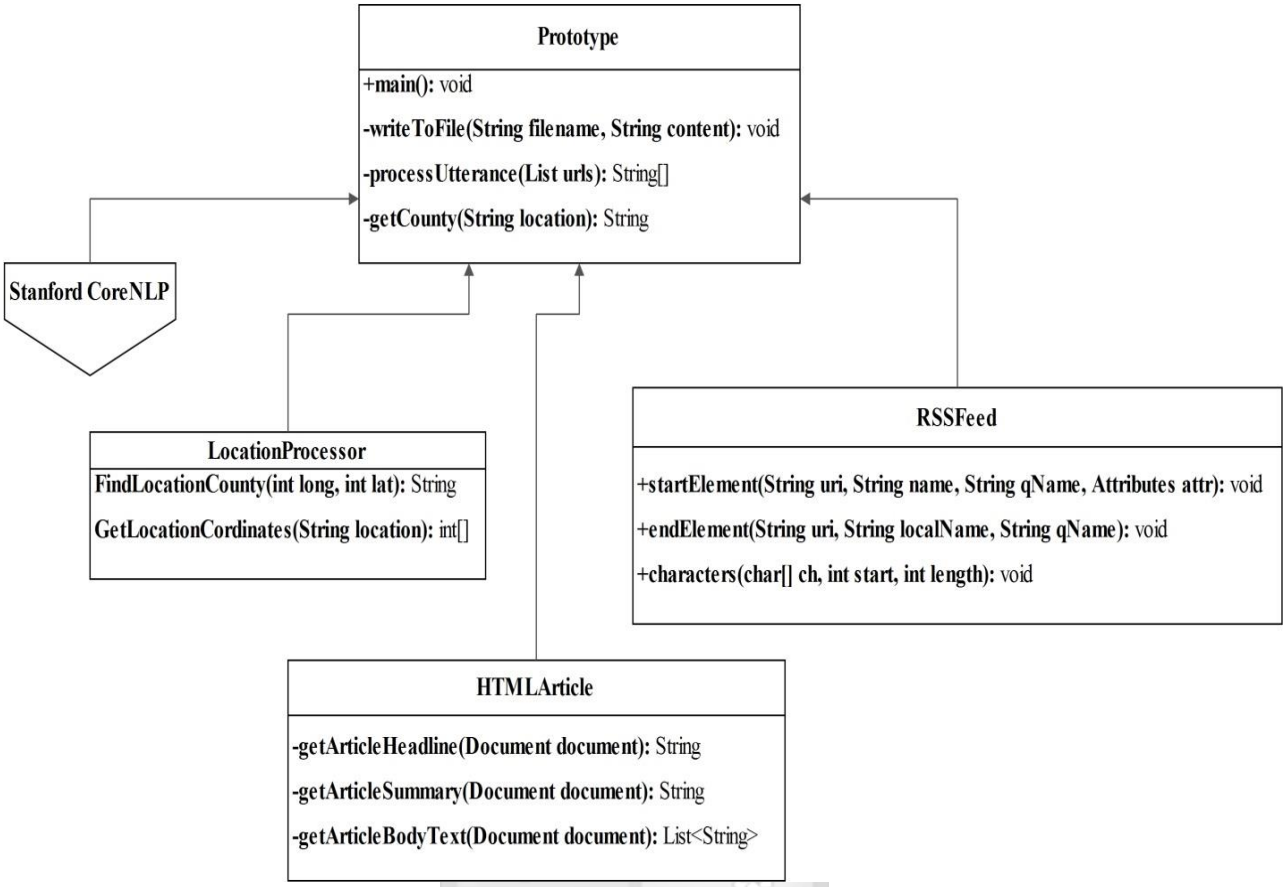


Figure 6.1: Improved prototype design to enhance HTML and RSS parsing functions.

The Stanford CoreNLP tool is constantly being improved to address shortcomings such as the ones mentioned before. The latest version allows for additional training words to be included to it and thus the system can be updated to learn new name entities. However with more custom requirements, using a less specialized tool such as Python NLTK would allow for greater customization but would require more development input. Ideally, a combination of the two could be used for better results as was done in this research. For the event detection module, various proposals have been put forward in how to apply machine learning for topic analysis and event prediction. Various topic modelling techniques such as those described in Latent Dirichlet Allocation have addressed issues of modelling text corpora and collections of discrete data to find short descriptions of the members of a collection that enable processing of large collections

while preserving the essential statistical relationships that are useful for basic tasks such as classification, novelty detection, summarization, and similarity and relevance judgements (Blei, Ng, & Jordan, 2003).

6.3. Performance Analysis

Figure 6.2 shows a line graph of the number of iterations it took to get to the optimum point that minimized the cost function and the prediction accuracy of the learning module at the optima against the number of features. Figure 6.3 shows a line graph of the final cost at the optima that minimized the cost function against the number of features. From Figure 6.2 it can be seen that the prediction accuracy of the learning module tended to increase with an increase in the number of features. This increase however leveled off after a certain number of features until adding more features resulted in very little or no increase in the prediction accuracy. The number of iterations that the minimization function went through to get the optimization parameters however reduced and then leveled off as the number of features increased. The reduction could be attributed to better defined optima as the number of features increased. The minimization function was therefore able to use a greater learning rate and therefore got to the optima faster.

From Figure 6.3 the un-regularized cost function at the point that the cost function was minimized generally decreased. However the decrease was not a smooth one and it fluctuated. This general reduction was seen to match the general increase in the accuracy of the learning module.

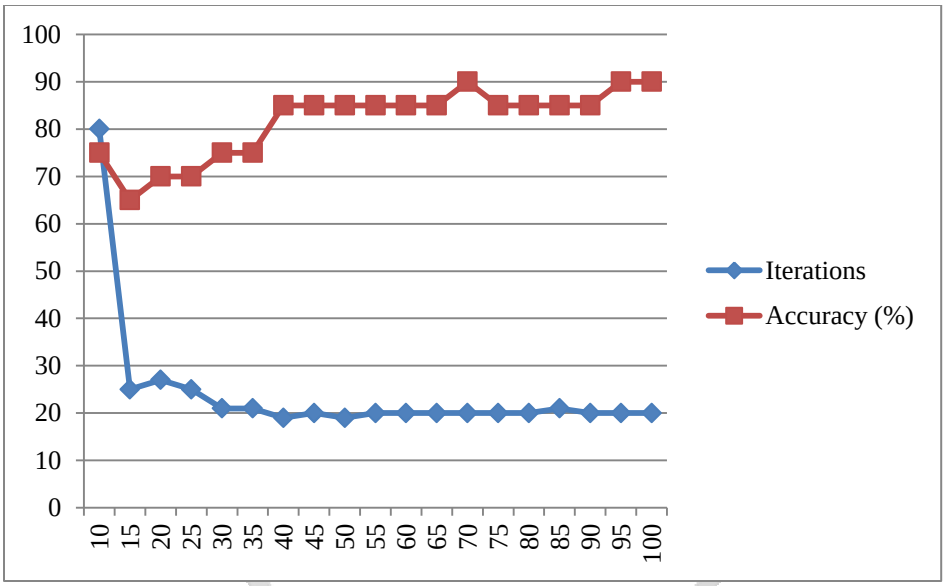


Figure 6.2: Line graph showing accuracy and number of iterations (y-axis) against number of features (x-axis).

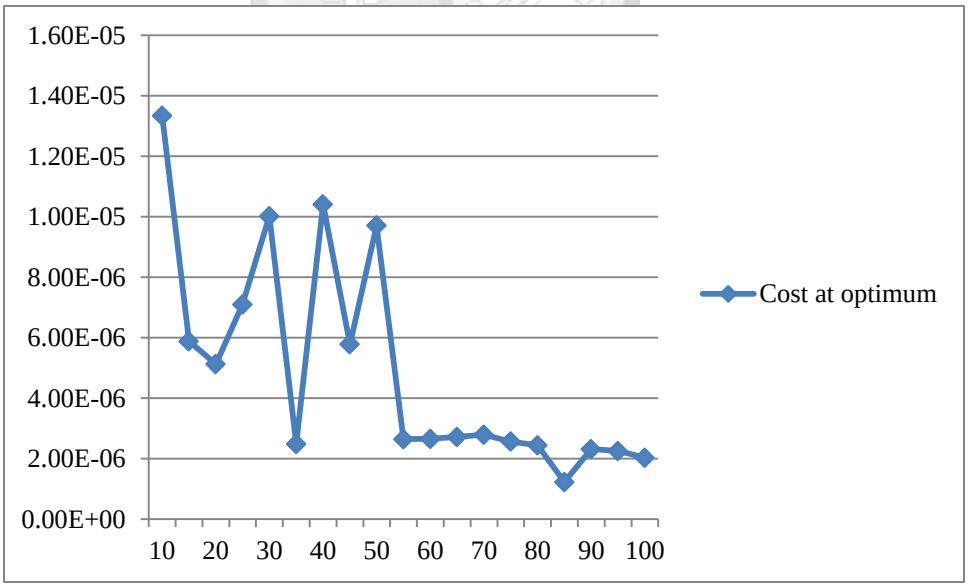


Figure 6.3: Line graph showing cost at optima (y-axis) against the number of features (x-axis).

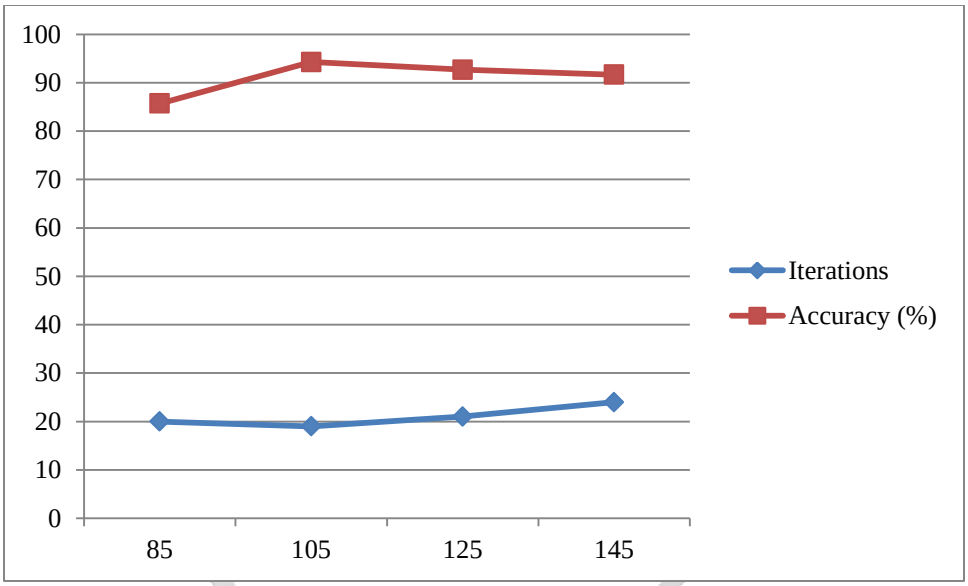


Figure 6.4: Line graph of iterations and accuracy with varying number of training examples.

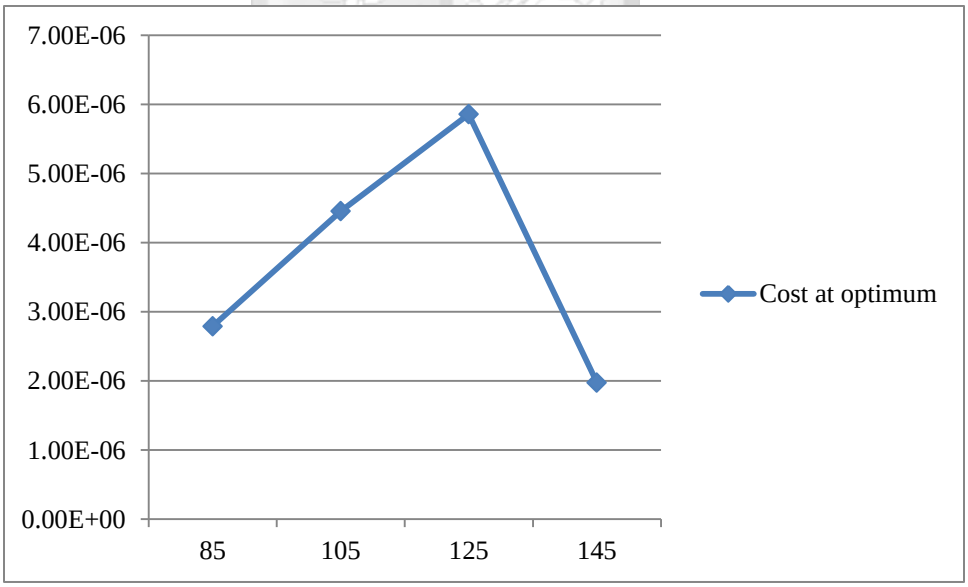


Figure 6.5: Line graph of the cost at optima against a varying number of training examples.

Figure 6.4 shows how the accuracy and number of iterations it took the minimization function to get the parameters that optimize the cost function varied with varying number of training examples. Both the accuracy and the number of iteration tendeds to increase with increasing number of training examples. This was also true of the cost function as shown in Figure 6.5. However a sudden drop in cost was observed at 145 training examples which may have been an anomaly within the training examples or the test examples.

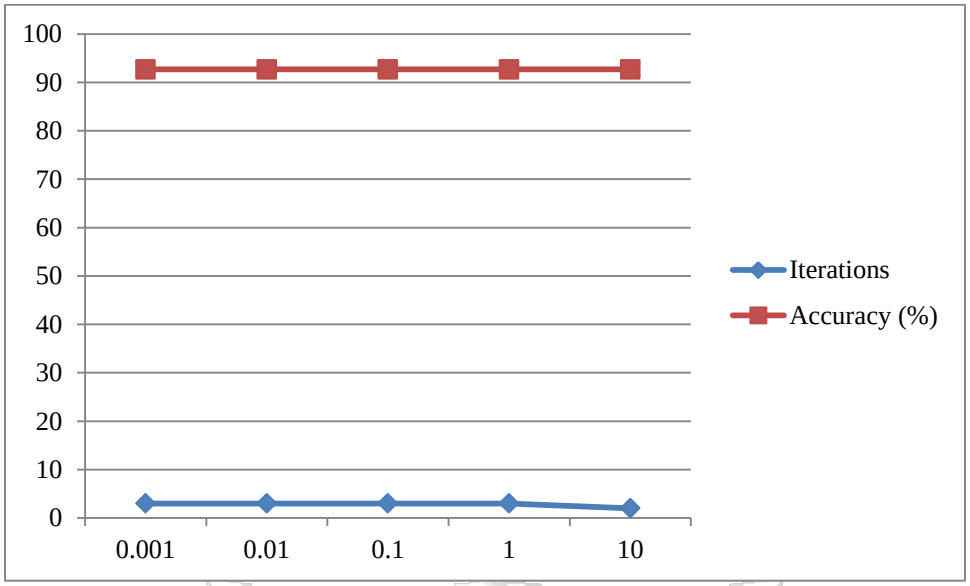


Figure 6.6: Line graph of iterations and accuracy with varying regularization factor.

Figure 6.6 shows that the accuracy and the number of iterations are not affected by regularizing the cost function. From Figure 6.7 however the cost sharply increases with an increasing regularization factor. Even a small regularization factor significantly increases the cost compared to the cost without a regularization factor. The final system would thus be implemented without regularizing the cost function since a regularization parameter significantly increased the cost without increasing the prediction accuracy of the learning module.

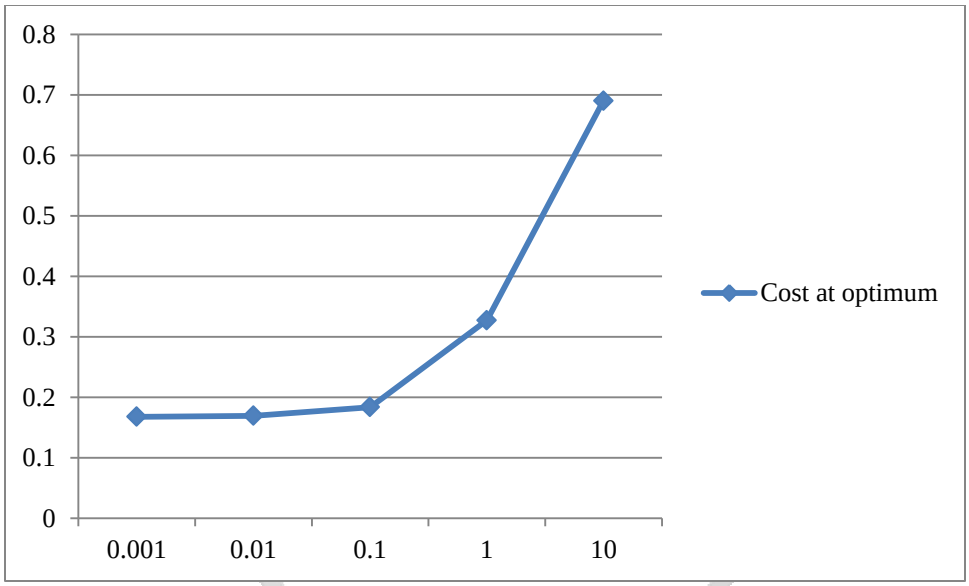


Figure 6.7: Line graph of the cost at optima against a varying regularization factor.

6.4. Comparison with other Systems

This research borrowed from other systems while trying to address shortcomings that may have existed in them. The prototype developed aimed at obtaining actionable intelligence from both present and past newswire content. The application of machine learning techniques while possibly more advanced in the other systems could be improved to reach and surpass the levels of information and event extraction in the other systems. A key goal in this research was to open up this area of research particularly when applied to Kenyan news content and this was achieved successfully. Other systems such as the EMM system apply a more black box approach which prevents other developers and researchers from looking into the logic that goes on behind the scenes of the working system. This shortcoming was addressed in this research by making all the algorithms and source code available for other researchers to modify and possibly improve on.

VT OMNES
VNVM SINT

Chapter 7: Conclusion and Recommendations

7.1. Conclusion

NLP has many applications and can be very useful when applied to newswire content. Online news has over the years become an important source of information that is free and easily accessible. While online news may sometimes not give an accurate report of all events that occur, it gives an adequate sample to study the links and relationships that exist between certain events and enables researchers to generate hypotheses that may lead to discovery of cause and effect relations human observers might ordinarily miss. This research will be useful to other researchers for a variety of reasons. Those researching on news trends will be able to gather insights into the topics of focus during certain periods and will be able to identify factors that influence news reports that are published and the events related to these reports. Organizations with interests in key events and statistics regarding the wellbeing of the country will be able to gather insight more easily without having dredge through massive amounts of data. Data scientists will also be able to generate reliable datasets more easily and efficiently.

This research was successful in identifying novel ways of visually representing information captured by using various NLP tools. This research also successfully implemented machine learning algorithms to detect events in articles. The prototype however could perform much better with improvements to the algorithms and with access to more news content. Many other tools exist that could have been used as alternatives to the tools used in this research. Graphical presentation tools like CanvasJS, D3, dycharts, and Google charts were considered as options for this research. Each would have resulted in a unique form of presentation with both advantages and disadvantages. While technical evaluation played a key role in this research, some intuition was also involved in determining what tools would best suit the tasks involved. In the same light, the choice of logistic regression over neural networks was done intuitively with the consideration of the level of mastery required to implement successful neural networks as well as the time that was available to satisfactorily conduct and complete the research. All these options while not included in this research were put into consideration for future research that would extend this research.

7.2. Recommendations

Implementing the prototype to a full scale working system would generate useful information on a variety of events. Some recommendations that would improve the functionality and performance of the prototype are;

i. For a full scale implementation it would be important to ensure that the system is designed to analyze all published articles in real time while eliminating repeated articles to avoid synthesizing redundant and inaccurate information.

ii. For the event detection module of the system, more training data from past articles should be collected to improve the prediction accuracy of the module as well as widen the scope of events that the module can detect.

iii. A multiclass classification design would enable the labeling of multiple events in a single training data set and would also allow the use of a single algorithm applied once to detect multiple events as compared to using a single class classification design and applying it to several different data sets for different events.

iv. Neural networks models have reportedly had more success in predictive learning models as compared to the other models. For this reason, replacing the logistic regression model in this research with a neural network model might improve the performance of the overall system.

v. Further insights into how variations of binary search trees and other data structures can be applied to reduce the computation speed of the system would be beneficial to help design a system that is both fast and accurate. An example is the Iacono's tree structure which has a significantly lower worst case running time than many other data structures.

vi. This research could have applications in online media particularly relating to news. Various organizations have made advancements into geographical mapping of events both locally and globally. This research further adds to those advancements with an additional focus on making the technologies involved open to other researchers.

vii. All of the source code that was produced during the implementation of this research was made available freely on the online code repository GitHub. Allowing other researchers and programmers to contribute and share insights on this research as well as their own would greatly expedite the progress being made in the fields of machine learning, natural language processing, among others.

viii. Involving the online community in the research process has also brought about great progress in various fields, among them machine learning and artificial intelligence. Crowdsourcing labeled news data would introduce efficiencies in the research process. Time spent collecting and labeling data could be channeled to other activities in the pipeline such as improving algorithms.

7.3. Suggestions for Future Research

For future research, more advanced learning algorithms could be applied to get more details pertaining to events. For a particular event such as an accident event, a lot more information could be captured, such as the types of vehicles involved, the number of fatalities or injuries, and the circumstances that led to the accident. For terror events, more detailed information such as the nationalities of the attackers, the method of attacks, and the nationalities of the victims of the attacks among others could be captured.

Application of models and algorithms such as Naïve Bayes, Markov models, and neural networks would greatly increase the possibilities in terms of the type of information that can be captured and would also greatly increase prediction accuracy. These models and algorithms could also be applied further to extracted information to investigate trends from events and topics that are reported over long periods of time. Future research could be done to investigate how news reports could be used to predict events before they happen. This kind of research would be instrumental in taking necessary steps to prevent occurrence of certain catastrophic events such as terror attacks.

Future research could also investigate novel ways of collecting and labeling data to use in training data sets. These methods could include crowdsourcing methods where members of online communities assist in labeling relevant news articles that can then be used by researchers in their work. Furthermore, the parsing of text from online news articles should be advanced to cover content from many more sources in order to gather as much information as possible to get more accurate data. The changing landscape of online news to include social media, blogs, and multimedia content should be also be factored in. Future research should therefore look into processing of audio content as well, that is, natural language processing of spoken language.

The possibilities for future research are numerous and it would be difficult to document all of them in a short amount of time. These possibilities should therefore be kept open for future researchers to delve into for the further advancement of the field.

References

Alber, S. M. (2010). *Toolkit for Action Research.* Blue Ridge Summit, PA, USA: Rowman & Littlefield Education. Retrieved from http://www.ebrary.com

Atkinson, M., & Van der Goot, E. (2009). Near Real Time Information Mining in Multilingual News. *Proceedings of the 18th International Conference on World Wide Web*, 1153-1154.

Best, C., Van der Goot, E., Blackler, K., Garcia, T., & Horby, D. (2005). *Europe Media Monitor-System Description.* EUR Report.

Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python.* O'Reilly Media Inc.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research 3*, 993-1022.

Bond, D., Bond, J., Oh, C., Jenkins, J. C., & Taylor, C. L. (2003). Integrated data for events analysis (IDEA): An event typology for automated events data development. *Journal of Peace Research*, 733-745.

Coreference Resolution. (n.d.). Retrieved March 29, 2016, from The Stanford Natural Language Processing Group: http://nlp.stanford.edu/projects/coref.shtml

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). Cambridge, Massachusetts: The MIT Press.

Dale, R., Moisl, H., & Somers, H. (2000). *Handbook of Natural Language Processing.* New York, NY, USA: CRC Press. Retrieved from http://www.ebrary.com

Dennis, A., Wixom, B. H., & Roth, R. M. (2012). *System Analysis and Design* (5th ed.). John Wiley & Sons, Inc.

Easterbrook, S., Singer, J., Storey, M.-A., & Damian, D. (2008). Selecting Empirical Methods for Software Engineering Research. In *Guide to Advanced Emprical Software Engineering* (pp. 285-311). Springer London.

Hogenboom, F., Frasincar, F., Kaymak, U., & De Jong, F. (2011). An overview of event extraction from text. *Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011). 779*, pp. 48-57. Tenth International Semantic Web Conference (ISWC 2011).

Jackson, P., & Moulinier, I. (2007). *Natural Language Processing for Online Applications: Text Retrieval, Extraction and Categorization.* Amsterdam, NLD: John Benjamins Publishing Company. Retrieved from http://www.ebrary.com

Jurafsky, D. (n.d.). Online lecture on Natural Language Processing, Introduction. Retrieved from http://www.youtube.com

Jurafsky, D., & Martin, J. H. (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (2nd ed.). Prentice-Hall.

King, G., & Lowe, W. (2003). An automated information extraction tool for international conflict data with performance as good as human coders: A rare events evaluation design. *International Organization*, 617-642.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. *52nd Annual Meeting of the Association for Computational Linguistics* (pp. 55-60). Baltimore: The Association for Computer Linguistics.

McCallum, A. K. (2002). *MALLET: A Machine Learning for Language Toolkit*. Retrieved September 2015, from http://www.cs.umass.edu/~mccallum/mallet

McClosky, D., Surdeanu, M., & Manning, C. D. (2011). Event extraction as dependency parsing. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:Human Language Technologies-Volume 1* (pp. 1626-1635). Association for Computational Linguistics.

Open Event Data Alliance. (2015). *PETRARCH Documentation*. Retrieved from Read the Docs: http://petrarch.readthedocs.org/en/latest/

Perkins, J. (2010). *Python Text Processing with NLTK 2.0 Cookbook : Use Python's NLTK Suite of Libraries to Maximize Your Natural Language Processing Capabilities.* Olton, Birmingham, GBR: Packt Publishing Ltd. Retrieved from http://www.ebrary.com

Piskorski, J., & Atkinson, M. (2011). Frontex real-time news event extraction framework. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 749-752). ACM. doi:10.1145/2020408.2020527

Piskorski, J., Tanev, H., Atkinson, M., & van der Goot, E. (2008). Cluster-centric approach to news event extraction. *New Trends in Multimedia and Network Information Systems*, 181-276. Retrieved from http://www.researchgate.net

Schrodt, P. A., Beieler, J., & Idris, M. (2014, March 23). Three's a Charm?: Open Event Data Coding with EL:DIABLO, PETRARCH, and the Open Event Data Alliance.

Schrodt, P. A., Davis, S. G., & Weddle, J. L. (1994, June 28). Political Science: KEDS - A Program for the Machine Coding of Event Data.

Schrodt, P. A., Gerner, D. J., & Yilmaz, O. (2004). Using Event Data to Monitor Contemporary Conflict in the Israel-Palestine Dyad. *International Studies Association*.

VT OMNES
VNVM SINT

Appendix A : Selecting Actions and Documentations

Research question - What kind of information is most useful in utilization of news media content?

Table A.1: Key actions and related documents for the first research question.

Key Actions	**Resulting Documentation**
News content analysis	✓ Analysis report.
Website traffic analysis	✓ Graphs and charts. ✓ Comparison tables.

Research question – What are the merits of the existing information capture and utilization techniques?

Table A.2: Key actions and related documents for the second research question

Key Action	**Resulting Documentation**
System testing	✓ Test report.
Reviewing research documents	✓ Literature review.

Research question – What are the challenges faced by the current information capture techniques and their utilization?

Table A.3: Key actions and related documents for the third research question

Key Action	**Resulting Documentation**
System testing	✓ Test report.
Reviewing publications	✓ Literature review.

Research question – How can information be extracted and utilized efficiently and effectively?

Table A.4: Key actions and related documents for the fourth research question

Key Action	**Resulting Documentation**
System analysis and design	✓ UML diagrams. ✓ Class diagrams.
Prototype development	✓ Prototype documentation.

Research question – What are the merits of the developed prototype?

Table A.5: Key actions and related documents for the fifth research question

Key Action	**Resulting Documentation**
Test prototype	✓ Test reports.
Data analysis and presentation	✓ Analysis reports. ✓ Graphs and charts.

Appendix B : Description of Actions

Research question - What kind of information is most useful in utilization of news media content?

Table B.1: Key actions and critical steps for the first research question

Key Action	**Critical Steps in Action**
News content analysis	✓ Reading through publish news articles over the duration of the research. ✓ Getting sample articles and labeling them by event.
Website traffic analysis	✓ Getting authorization to access web analytics for the Daily Nation website. ✓ Conducting a comprehensive analysis of web traffic according to keywords and topics.

Research question – What are the merits of the existing information capture and utilization techniques?

Table B.2: Key actions and critical steps for the second research question

Key Action	**Critical Steps in Action**
System testing	✓ Testing of existing systems. ✓ Identifying the practical advantages of existing systems.
Reviewing publications	✓ Analyzing publications on existing systems. ✓ Reviewing merits of the existing systems identified in the publications.

Research question – What are the challenges faced by the current information capture techniques and their utilization?

Table B.3: Key actions and critical steps for the third research question

Key Action	**Critical Steps in Action**
System testing	✓ Testing of existing systems. ✓ Identifying the practical disadvantages of existing systems.
Reviewing publications	✓ Analyzing publications on existing systems. ✓ Reviewing demerits of the existing systems identified in the publications.

Research question – How can information be extracted and utilized efficiently and effectively?

Table B.4: Key actions and critical steps for the fourth research question

Key Action	**Critical Steps in Action**
System analysis and design	✓ Identify requirements for a prototype to be developed. ✓ Design a prototype
Prototype development	✓ Develop a prototype.

Research question – What are the merits of the developed prototype?

Table B.5: Key actions and critical steps for the fifth research question

Key Action	**Critical Steps in Action**
Test prototype	✓ Conduct various tests on the prototype. ✓ Identify merits of the designed prototype.
Data analysis and presentation	✓ Analyze data collected during the research. ✓ Present the data in a clear and logical manner.

VT OMNES
VNVM SINT

Appendix C : Content Usage Agreement

Nation Media Group Limited

Nation Centre
Kimathi Street
Post office Box 49010 G.P.O
Nairobi 00100
Kenya
Telephone: 3288000/1/2/337710/221222/211448
Telefax:2214565/213946/313332/2219882
Email:customercare@nation.co.ke
Website:http://www.nationmedia.com

Dear Mr. Mbuthia,

We refer to your request for permission to access and or reproduce content from Nation Media Group Websites (Hereinafter jointly called "the Websites") in the proposed Dissertation you are required to submit in pursuit of a degree you are pursuing at the Strathmore University.

This is to confirm that Nation Media Group Limited is willing to give to you as author of the proposed Dissertation the License and permission to access reproduce and publish the contents of the Websites in the said Dissertation upon the following terms and conditions:

1. That the permission given to you shall be limited to the research of the said Dissertation and does not constitute a transfer or assignment of Nation Media Group's copyright and other intellectual property rights in the said Articles.
2. That the permission to access to and use of the content of the Websites shall be reserved strictly for use in the Dissertation and may not be transferred to or shared with any other person without the written permission of Nation Media Group Limited.
3. That the ***Nation Media Group's respective websites*** shall be duly acknowledged as the medium in which the content was published and the respective reporters and authors of the articles shall be equally acknowledged as such.
4. That the content of the Websites shall be used and cited in the same format as they were first published in the Websites and may not be altered, distorted or otherwise changed in any way whatsoever;
5. That the License and permission hereby granted to you shall be shall be subject to the Copyright laws of Kenya.
6. That in the event of breach of any of the above conditions, Nation Media Group Limited shall be entitled to revoke this permission.

Please confirm your unconditional acceptance of the above-mentioned conditions by signing and returning to us the enclosed copy of this letter as soon as possible.

Signed : SAMUEL KAMAU MBUTHIA

Date : 05/11/2015