2020

Sentiment analysis model for detection of radicalization on twitter.

Oluoch, Emmanuel Olang'
Faculty of information technology
Strathmore Universit

Sentiment Analysis Model for Detection of Radicalization on Twitter

By

Emmanuel Oluoch Olang'

114899

A Research Thesis Submitted to the Faculty of Information in partial fulfilment of the requirements for the award of the degree of Master of Science Information Technology (MSc. IT) at Strathmore University.

Master of Science Information Technology

Strathmore University

July 2020

Declaration and Approval

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

.....

.....

.....

Approval

The thesis of Emmanuel Oluoch Olang' was reviewed and approved by the following:

Dr. Joseph Orero
Dean, Faculty of Information Technology
Strathmore University

Dr. Joseph Orero
Dean, Faculty of Information Technology
Strathmore University

Dr. Benard Shibwabo,
Director of Graduate Studies,
Strathmore University

Abstract

In the recent past, radical acts such as terrorist attacks on highly populated areas have become a major security issue in Kenya hence there is a constant fear of becoming a victim of violent acts perpetrated by individuals who are radicalized before carrying out such harmful acts. Current efforts by the security organs used to detect radicalization involve monitoring public communications channels, relying on information gathered from community policing, random suspect searches, and other intelligence services. However, these approaches have a set of drawbacks first being the manual human intervention needed in decision making even in cases where technology has been used to such as the use of web crawlers. In addition to this, automated text classification techniques rely on feature generation techniques that don't take into account the context of the text and that are also subjected to sparsity when classifying long texts. On the other hand, to grow membership numbers, radical groups use the public platforms offered by social media such as Twitter to disseminate radical ideologies and facilitate the recruitment of those who support such ideologies.

In this study, I propose the use of sentiment analysis model to detect online radicalization. The model uses text classification using artificial neural networks to learn word relations within a corpus and generate corresponding features represented as lower-dimensional vectors. By using Continuous Bag of Words (CBOW) encoding and Word2Vec methods, tokens were represented as integers and an embedding of all corpus words was generated i.e trained to be used in one of the layers of the classifier's neural network. Using the embeddings, labeled data instances, a four-layered recurrent neural network classifier was developed for text classification of radical and non-radical statements. The classifier then uses the pre-trained embeddings to refer corresponding vector values for tokens within the input padded sequence hence classify the new instance and return a rounded float value of 1 or 0 indicating a class. To train the model, I used pre-existing data of tweets collected from Twitter using keyword guides and also tested using data from the Kenyan setting. The model developed had an accuracy score of 95% after varying iterations of training, testing, and validation.

Keywords: Radicalisation, Social Networking; Natural Language Processing

Table of Contents

Declaration and Approval ii

Abstract iii

List of Tables viii

List of Figures ix

List of Abbreviations xi

Chapter 1 Introduction 1

1.1 Background of Study 1

1.2 Problem Statement 2

1.3 Aim 2

1.4 Research Objectives 3

1.5 Research Questions 3

1.6 Justification 3

1.7 Scope and Limitation 3

Chapter 2 Literature Review 4

2.1 Introduction 4

2.2 Methods of Detecting Radicalization in Kenya 4

2.2.1 Intelligence Services 4

2.2.2 Community Policing 4

2.2.3 Random Searches 5

2.3 Online Radicalization 5

2.3.1 Link Based Bootstrapping (LBB) 6

2.3.2 Text Classification Techniques 7

2.4 Machine Learning for Text Classification 7

2.4.1 Artificial Neural Networks 10

2.4.2 Support Vector Machines 12

2.4.3 Naive Bayes 14

2.4.4 Text Classification Process 14

2.5 Related Work 19

2.5.1 Sentiment Analysis Using Polarity Dictionaries 19

2.5.2 Twitter Sentiment Classification using N-grams 19

2.5.3 Twitter Trending Topic Classification 19

2.5.4 Real-time sentiment analysis for the detection of terrorist activities in Kenya 20

2.6 Proposed Approach 20

2.7 Conceptual Framework 21

Chapter 3 Research Methodology 22

3.1 Introduction 22

3.2 Research Design 22

3.3 Datasets and Data Collection 22

3.3.1 Kaggle Data Repository 23

3.3.2 Sentiment 140 Project 23

3.3.3 Mining Data from Twitter 23

3.4 Dataset Pre-processing 24

3.5 System Development Methodology 24

3.5.1 Phases of RAD 25

3.5.2 Reasons for choosing RAD 26

3.6 Research Quality 26
3.7 Performance Visualisation 27
3.8 Model Validation 27
3.8.1 Experiment 1: Using Alternative Machine Learning Algorithms 27
3.8.2 Experiment 2: Using Different Word Vocabularies 27
3.9 Ethical Considerations 28
Chapter 4 Design and Architecture 29
4.1 Introduction 29
4.2 Design 29
4.3 Functional Requirements 29
4.4 Non-Functional Requirements 30
4.4.1 Persistent Storage 30
4.4.2 Data Integrity 30
4.4.3 Scalability 30
4.5 Architecture 30
4.6 Use Case Diagram 32
4.6.1 Use Case Diagram Description 33
4.7 Sequence Diagram 34
4.8 Context Diagram 35
4.9 Level 0 Data Flow Diagram 35
Chapter 5 Implementation and Testing 37
5.1 Corpus Creation 37
5.1.1 Data Collection 37

5.1.2 Data Pre-processing 37
5.1.3 Data Labelling and Merging 39
5.2 Model Training 40
5.2.1 Model Architecture 40
5.2.2 Creating Word Embeddings 40
5.2.3 Training the Model 42
5.3 Model Validation 43
5.4 Using the Model in Prediction 45
Chapter 6 Experiments and Discussions 47
6.1 Introduction 47
6.2 Experiments 47
6.2.1 Experiment 1: Using Different Classifiers 47
6.2.2 Experiment 2: Using Different Word Vocabularies 48
6.2.3 Experiment 3: Using Kenyan Data 48
6.2.4 Classification Comparisons 50
6.3 Discussions 52
6.3.1 Developed Model 52
6.3.2 Validation Experiment Results 52
Chapter 7 Research Conclusions and Recommendations 53
7.1 Conclusion 53
7.2 Recommendations 54
7.3 Future Works 54
References 55

List of Tables

Table 2.1: Structure of a Confusion Matrix 8

Table 2.2 BoW Presentation 17

Table 5.1: Confusion Matrix 43

Table 5.2: Classification Values 43

Table 5.3 Performance of the RNN (LSTM) Model 44

Table 6.1: Test Classification Results 51

List of Figures

Figure 2.1: Structure of a Machine Learning Algorithm (Brownlee, 2019) 8

Figure 2.2: Artificial Neural Network Structure 10

Figure 2.3: LSTM Block (Rathor, 2018) 11

Figure 2.4: LSTM Gates (Rathor, 2018) 12

Figure 2.5: Optimal Hyperplane and Support Vectors 12

Figure 2.6: SVM Margins 13

Figure 2.7: Text Classifier Development (Geitgey, 2018) 14

Figure 2.8: Text Classification Process 15

Figure 2.9: Pre-processing Stages 15

Figure 2.10: Conceptual Framework 21

Figure 3.1: Phases of RAD 25

Figure 4.1: System Architecture 31

Figure 4.2: Use Case Diagram 32

Figure 4.3: Sequence Diagram 34

Figure 4.4: Context Diagram 35

Figure 4.5: Level 0 Data Flow Diagram 36

Figure 5.1: ISIS Fan-Boy Tweets Data 37

Figure 5.2: Sentiment 140 Data 38

Figure 5.3: Data Cleaning Function 39

Figure 5.4: Proposed RNN(LSTM) Model Architecture 40

Figure 5.5 Learning Embeddings For-Loop 41

Figure 5.6: Testing Word Similarities 41

Figure 5.7: Model Summary 42

Figure 5.8: Confusion Matrix Plot 43

Figure 5.9: Model Accuracy Visualization 44

Figure 5.10: Model Loss Visualization 44

Figure 5.11: Saving and Loading LSTM Model 45

Figure 5.12: Sample String Test Cases 45

Figure 5.13: Test Case Float and Integer Results 46

Figure 6.1: Experiment Two Performance 48

Figure 6.2: Experiment Three Corpus 49

Figure 6.3: Experiment Three Performance 50

List of Abbreviations

ANN	Artificial Neural Network
API	Application Programming Interface
BoW	Bag of Words
DFD	Data Flow Diagram
IDF	Inverse Document Frequency
HTTP	Hyper-Text Transfer Protocol
JSON	Java-Script Open Notation
LBB	Link Based Bootstrapping
LSTM	Long Short-Term Memory
OAuth	Open Authentication
RAD	Rapid Application Development
RNN	Recurrent Neural Networks
SVM	Support Vector Machine
TF-IDF	Term Frequency – Inverse Document Frequency
UML	Unified Modelling Language
URL	Uniform Resource Locator
VSM	Vector Space Model

Chapter 1 Introduction

1.1 Background of Study

Militant groups such as ISIS, Taliban, Al-Shabaab, Boko-Haram have long used the internet and social media for communication, information gathering, recruitment, and social network engagement (Almagor, 2012). Much of this content is propaganda, meant to inspire recruits, demonstrate to investors that their money is being well spent, and shore up morale among existing followers. Social media also is a way to involve part-time supporters who can modify Islamic State images or generate their content and disseminate it. Such propaganda also is used by face-to-face recruiters who reinforce their in-person sessions with video or teaching distributed through the internet.

Social media is defined as a collection of online applications that provide its users with a platform to create and share content (Zarella, 2009). Twitter is one of the free popular microblogging services where users create status messages posts (called “tweets”) of up to 280 characters in length (Go, Bhayani, & Huang, 2009). These tweets express opinions about different topics in given geo-locations.

Over the past decade, there has been increased growth of penetration of mobile devices leading to easier access to the internet (Almagor, 2012). Radical groups have taken advantage of micro-blogging sites and social media networks to facilitate their communications as it is an easier means to reach the masses and impact people's opinions. It is crucial to identify this type of communication and have thus keep track of various online activities performed by radical groups such as radicalization and recruitment (Archetti, 2015).

Sentiment Analysis is a technique that uses Natural Language Processing and Machine Learning techniques to extract, identify, and characterize the sentiment of a text set (Gupta, 2018). It is also defined as the process of computationally identifying and categorizing opinions expressed in a set of texts to determine whether the writer's attitude towards a particular topic is positive, negative, or neutral. It is not to be confused with Sentiment Mining which focuses on calculating` the general opinion of a particular group of people, extracting predefined keywords to check their frequency and perform statistical algorithms. When teamed up with Machine Learning, it can train itself to identify meanings from the text along with newly discovered words and patterns. Thus, the results of sentiment analysis get better with the amount of text it processes.

1.2 Problem Statement

Existing methods employed in the detection of online radicalization suffer from a variety of limitations, thus prove inefficient as a means of detecting radicalization. These limitations can be grouped into two broad categories first being the reliance on human interpretation in the detection processes when classifying instances which makes the overall detection process time consuming given the enormous amount of unstructured data obtained from social media sites. The second limitation of the existing system originates from using word count for corpus feature generation hence a document may be wrongly classified as the technique doesn't take into consideration how the wording in a document relates with other words in the same document; additionally, this leads to a high dimensional feature vector when working with long texts.

This research demonstrates the effectiveness of context consideration in text classification by introducing artificial neural networks to learn word relations within a corpus and generate corresponding features represented as lower-dimensional vectors i.e. an embedding where words that have the same meaning have a similar representation. Using these features and labeled data, a machine learning classifier is then developed using a Long Short-Term Memory recurrent neural networks alongside natural language processing techniques. The use of embedding as features for machine learning is aimed at improving classification accuracy by addressing the TF-IDF drawback of only considering word count for classification and ignoring the semantic similarities between words in a statement (Chawla, 2017).

This automatic classification will help in detecting radical groups specifically when they use the online platform to carry out radicalization. Overall this is a faster and proactive solution towards addressing the security threat posed as a result of radicalization.

1.3 Aim

This research aimed to develop a text classifier model that can be used for detecting radical texts and thus help identify radicalization on social media network Twitter caused as a result of interaction with radical individuals and groups on the online platform. The model can be integrated into the various system as per the end-user architecture.

1.4 Research Objectives

i. To investigate problems associated with the existing methods used in detecting radicalization.
ii. To review existing data mining and machine learning approaches available for use in detecting radicalization on social media networks.
iii. To propose a machine learning model that can be used to detect radicalization on social media.
iv. To test and validate the proposed model.

1.5 Research Questions

i. What are the existing techniques used in the detection of radicalization on social media?
ii. What are the current machine learning techniques applied in the detection of radicalization on social media?
iii. What is the best way to design the proposed model?
iv. How will the developed model be tested and validated?

1.6 Justification

The objective of this research was to come up with a model that will help in the identification of radicalization messages expressed on the social media network, Twitter in the form of publicly visible tweets. Given the growing amounts and availability of social media data, this research took into account how words relate to each other in a document by using the embedding approach for feature representation which takes advantage of large amounts of data. The developed model would thus provide a foundation for detecting radical groups with minimum human involvement in the detection process. This is aimed to benefit security organs by having an accurate and fast means for detection as compared to pre-existing methods.

1.7 Scope and Limitation

This study limited its analysis to detecting radicalization on the social media platform Twitter. The use of memes, audio, and video within tweets was not considered. Techniques of offline communicative behaviors by measuring levels of stress or anxiety, or detecting patterns associated with deceit will not be benefited directly by this research.

Chapter 2 Literature Review

2.1 Introduction

As described in the previous chapter, the goal is to propose a model for detecting radicalization on the social media network: Twitter. The first part covers the methods of detecting online radicalization as well as an overview of the processes involved in radicalization. The second section covers machine learning methods for sentiment analysis and the process of text classification. The section reviews additional works in the field of sentiment analysis. Finally, the last section highlights the proposed approach based on the literature reviewed.

2.2 Methods of Detecting Radicalization in Kenya

This section describes identified local techniques used by security organs detecting radical groups locally as well as detecting online radicalization. Since the Al-Qaeda 1998 bombing of the U.S. embassy in Nairobi, which killed more than 220 people Kenya has since become a victim of terrorist attacks mostly linked with the Al Shabab terror group. Following the launch of the Kenya National Strategy to Counter Violent Extremism, launched by H.E. the President in late 2016 (Magogo, 2017), the government aimed at tackling the threat of terrorist groups using the following avenues:

2.2.1 Intelligence Services

Aimed at detecting possible threats by going through available intel from various sources then informing relevant security organs. In Kenya, this is done via the National Intelligence Service (NIS) which is one of the subbranches of the Intelligence Agency (Githinji, 2020). Other certified aginces also aim to provide the information gathering capabilities but this often results in all parties involved in inteligencegathering operating in a rivalry manner (Blair, 2013).

2.2.2 Community Policing

This practice pioneered by the government aims to expand the partnership of government agencies and civil society. The approach mainly involves the community to participate in improving security by using informants and general citizens to keep track of those within their small-sized neighborhoods (Community Policing, 2020). The Kenyan Police force has constantly requested and relied on this sort of civilian aid/assistance to aid in quicker acces to information that may be used against various militant groups especially along the nations borders (Yusuf, 2020).

2.2.3 Random Searches

Involve checking a subject's general documentation e.g. travel documents, crime history, affiliations, etc. This is mostly done on persons highly suspected to be security threats. This is mostly based on the information gained from the above tow approaches and mainly used to get information without causing any alerts that may render the process (Xuequan, 2010). Due to dependency an actual physical searches, this technique has at times been persieved as harsh with most suspected indiviudals claiming harassment (Namwaya, 2016).

2.3 Online Radicalization

Online Radicalisation is a gradual process in which a person or group follows a radical philosophy which may lead to an increased willingness to support or use violence for political purposes. The cycle of radicalization is special to every person and appears to include a mixture of common cognitive and behavioral characteristics, systemic grievances, politicized by a unifying philosophy or a rallying cause that promotes a "de-pluralization cycle", a term that defines how a person becomes ever more narrow-minded in relation to key political ideas and values (Delphine & Camille, 2020).

Much has been said since the 9/11 twin tower attacks on how technologies such as the Internet and global communication networks have various roles in sustaining transnational terrorism, spreading their ideology and recruiting activities (Archetti, 2015). In the year 2015, an average of 90,000 Twitter accounts was suspected to support some sort of extremist/radical group ideology (Saul, 2015). A popular tactic used by radical groups is what's referred to as a *Media Mujahideen* which is a group of people fighting on media platforms to promote extremist propaganda (Bartlett & Fisher, 2015). To mark different topics, segments, and the popularity and relevance of a tweet a word phrase commonly referred to as a hashtag is used. The Islamic State of Iraq/Levant, also known as the Islamic State, ISIS is one major group globally identified as being radical/extremist (Security Council Adopts Resolution , 2014). ISIS uses hashtags campaigns showing support for ISIS and its cause, these included: #ILOVEISIS, #ALLEYESONISIS, #IS- LAMICSTATE (AFP, 2014).

Since the year 2011, some jihadist member forums have distributed robust media strategies that aim at the development of social media Mujahideen. Such strategies recommend accounts to follow on social media and how to use social media in their fights (Fisher & Prucha, 2014). A guide developed as a result of these strategies is the: *"The Twitter Guide: The Most Important Jihadi Sites and Support for Jihad and the Mujahideen on Twitter"* that outlines the main reasons for using the Twitter platform and states how it is an important arena of the

electronic front. Much of the current contact between radical groups and potential recruits is through the internet. The ability to recruit individuals from abroad makes radical groups quite dangerous as the recruits can carry attacks within their location or travel to target locations without even meeting their recruiters (Turner, 2018). Two of the common strategies used by radical groups during recruitments are (Turner, 2018):

i. The Role Model: An exceptional individual is portrayed on online platforms and potential recruits expected to follow in his/her path given their exceptional role model nature.
ii. The Sister/Brother: A potential recruit is approached by an individual pretending to have had similar life experiences hence appear as a sister/brother for the potential recruit to confide in during difficult times.

Apart from the above-mentioned recruitment strategies, there exist other approaches that are much more specifically tailored towards radicalization. A report on radicalization by the United Nations Office on Drugs and Crime UNODC (UNODC Education for Justice, 2018) mentions some of these approaches as:

i. The Net: violent radical and terrorist organizations distribute undifferentiated propaganda to a target audience considered homogeneous and susceptible to propaganda, such as video clips or tweets.
ii. The funnel: includes a gradual strategy to target individual individuals considered ready for recruitment, using psychological strategies to improve commitment and dedication. Even targeted kids who are resistant to full recruitment will build optimistic outlooks on group activities and,
iii. Infection: when it is impossible to enter the target demographic, a 'player' may be deployed to seek recruitment from within, using direct and personal appeals. Appealing to grievances, such as marginalization or social dissatisfaction, can reinforce the social ties between the recruiter and the targets.

The techniques used in detecting online radicalization are summarized into two depending on the original sources of the data used. These two are the Link Based Bootstrapping approach and the Text Classification Techniques.

2.3.1 Link Based Bootstrapping (LBB)

Link-based bootstrapping techniques are among the most commonly used to identify radical material online. The technique concentrates on various Internet modalities, including websites, forums, and online forums. Next, they classify a collection of seed URLs from

authoritative sources. The URLs are then expanded using backlink search and favourite links to accumulate related URLs. LBB assumes that extremist websites link to each other and form some sort of a bigger group structure. Web crawlers are used to manually download, collect, and analyze the content (UNODC Education for Justice, 2018).

2.3.2 Text Classification Techniques

Text classification techniques revolve around classifying texts either manually or automatically as being radical or not. The techniques rely heavily on the use of data from popular chat forums and sites known to be used by radical groups for carrying out radicalization. Multiple publications have been written to describe the various implementations of this technique.

A paper written by Weber and Darwish describes the use of a bag of words feature representation approach in building a binary classifier aimed at predicting future support or opposition for ISIS (Weber, Darwish, & Magdy, 2016). This approach also made use of Twitter data i.e. individual terms, hashtags, and user mentions. At a personal, historic level, the researchers managed to predict future support or opposition of a user for ISIS with 87% accuracy using an SVM based classifier.

Another research by (Huang, Fu, & Chen, 2010) presents alternative implantation of the text classification technique that can be used as a text-based video content classifier for use on online video sharing sites. They create a testbed of 224 positive and negative videos with text titles, manually tagged by to evaluate their approach. They used four feature sets: content-specific features(word and character n-grams, video tags and categories), lexical(character-based, word-based), syntactic(frequency of function and punctuation words), and feature selection strategy (using Information Gain) as training data for their classifiers. They report the best accuracy results on an SVM classifier with the content-specific features.

2.4 Machine Learning for Text Classification

Machine learning as a branch of artificial intelligence that systematically applies algorithms to synthesize the underlying relationships among data and information (Mariette & Khanna, 2015). These algorithms alongside other tools help develop a machine learning model that relies on mathematical computations and some example inputs (training data) and to make some predictions or decisions. With the model, an input can be mapped to a range of outputs depending on a given set of rules or conditions.

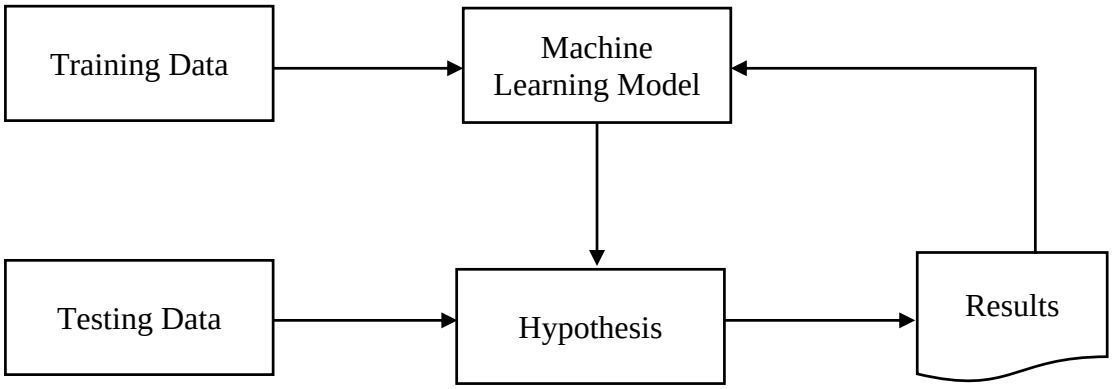


Figure 2.1: Structure of a Machine Learning Algorithm (Brownlee, 2019)

The goal is to develop relevant models that learn and perform without human assistance. Instead of static programming that tells the computer what to do, this approach will construct algorithms that can learn from the provided data.

Sentiment Analysis is a technique that uses Natural Language Processing to extract, identify and characterize the sentiment of a text set (Gupta, 2018). The main machine learning methods used in sentiment analysis fall under Supervised Learning since the data used developing a model has a label appended to it to indicate its polarity/sentiment or class. To determine how well a method performs, the results from a supervised learning approach are normally visualized in what is called a confusion matrix, also known as an error matrix. It is a two by two matrix containing the number of false negatives, false positives, true positives, and true negatives (Stehman, 1997).

Table 2.1: Structure of a Confusion Matrix

	Predicted (0)	Predicted (1)
Actual (0)	True Positive (TP)	False Negative (FN)
Actual (1)	False Positive (FP)	True Negative (TN)

i. True positive: Instances correctly predicted as Positive i.e. 0.
ii. True negative: Instances correctly predicted as Negative i.e. 1.
iii. False-positive: Instances wrongly predicted as Positive, yet the true class should be Negative.
iv. False-negative: Instances wrongly predicted as Negative, yet the true class should be Positive.

The four values obtained from the confusion matrix can then be used to calculate various performance measurements of a machine learning model used for text classification (Upender, 2018). These measurements are:

The accuracy of the model

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

The precision of the model i.e. the ratio of correctly classified tweets to the total number of tweets classified under a particular category (Feldman & Sanger, 2006) is given by:

$$Precision = \frac{TP}{TP+FP}$$

The recall of the model i.e. the number of correctly classified tweets among all tweets in belonging to a given category (Feldman & Sanger, 2006) is given by:

$$Recall = \frac{TP}{TP+FN}$$

The F-Score i.e. the weighted harmonic average of the test's Precision and Recall (Feldman & Sanger, 2006), is given by:

$$F - Score = \frac{Precision\ x\ Recall}{Precision+Reacll}\ x\ 2$$

With regards to text classification, specifically sentiment analysis, one can use the classical machine learning algorithms or the modern deep learning-based algorithms to develop a text classifier. The following section covers the implemented machine learning algorithm as well as alternative classical algorithms commonly used in the field of sentiment analysis.

2.4.1 Artificial Neural Networks

Artificial Neural Networks are a set of algorithms that closely resemble the human brain and are designed to recognize patterns by analyzing and processing information received from the input layer (Mittal, 2019). In a given network there are two main layers i.e. the Input and Output layers. Also, on the network exists neurons which are tiny units arranged in a series of layers. The figure below shows a general artificial neural network structure.

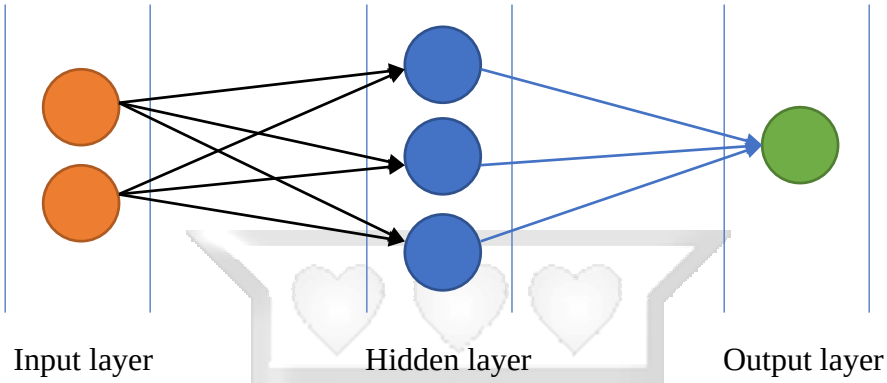


Figure 2.2: Artificial Neural Network Structure

The input layer is designed to receive different forms of information world and then recognize, interpret and classify. The output layer sits on the opposite end of the network awaiting the result of the process. In between the input and output layers are hidden layers that perform most of the work determining how to process the information fed in as inputs. The connections between one layer and the next are known as weights and can be either positive or negative. A layer receives inputs from the layers to its left, and the inputs are multiplied by the weights of the connections.

With the growth machine learning techniques being used in automation of most processes including text classification, there have been works done to illustrate how this can be achieved. An approach of knowledge-enhanced neural networks for sentiment analysis (Chen, 2019) demonstrates how existing sentiment analysis systems aren't suitable in identifying aspect-opinion pairs and propose a novel framework approach using artificial neural networks to address the identified gaps. Another article on automated text classification by (Ghiassi, Arnuado, & Moon) describes how current classification techniques rely mostly on Naïve Bayes, k-Nearest-Neighbor (kNN), Support Vector Machines (SVM) and propose how a dynamic artificial neural network can be used as scalable classification strategy without always having to set classification parameters.

There exists a variety of implementations of an artificial neural network such as (Tch, 2017): Feed Forward, Recurrent Neural Network, Convolutional Neural Network, Generative Adversal Network, and much more. For the proposed approach of this research, the Recurrent Neural Network specifically its Long Short-Term Memory implantation was the most suitable option given the text classification problem at hand.

A Recurrent Neural Network can be defined as a feed-forward network with a memory component within for storing the output from layers to be reused in the input (Mittal, 2019). A feed-forward network has the data move in only one forward direction from the input layer until it reaches the output layer. The sum of the products of the inputs and weights is calculated during data movement and the result passed to the output layer for processing and presentation of results. The term recurrent implies the network performs the same function for every input of data while the output of the current input depends on the past one computation (Mittal, 2019).

An LSTM network is a modified version of recurrent neural networks, which makes it easier to remember relevant data in memory by providing a solution to the fading gradient problem experienced when using feed-forward RNN architectures (Olah, 2015). LSTMs are designed to avoid the long-term dependency problem by choosing what to remember as data moves along the neural network. In a given network an LSTM block has three inputs and two outputs depicted in Figure 2.4.

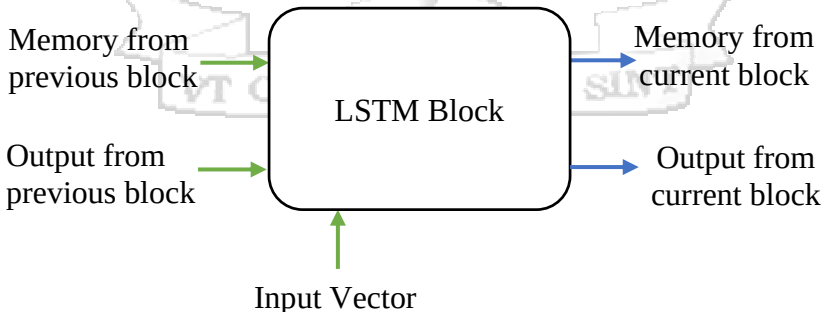


Figure 2.3: LSTM Block (Rathor, 2018)

To regulate information flow and change of memory contents, LSTMs use a series of three gates within each block, two vector operations (Element Wise Multiplication and an Element Wise Summation), two activation functions i.e. Sigmoid and Tanh function. The three LSTM gates handle various functions within the block i.e. By using the sigmoid function, the, forget gate decides which information to be omitted from the cell in that particular timestamp. The

input gate then decides how much of this unit is added to the current state while the output gate determines which part of the current cell makes it to the output to be used as one of the inputs in the next block.

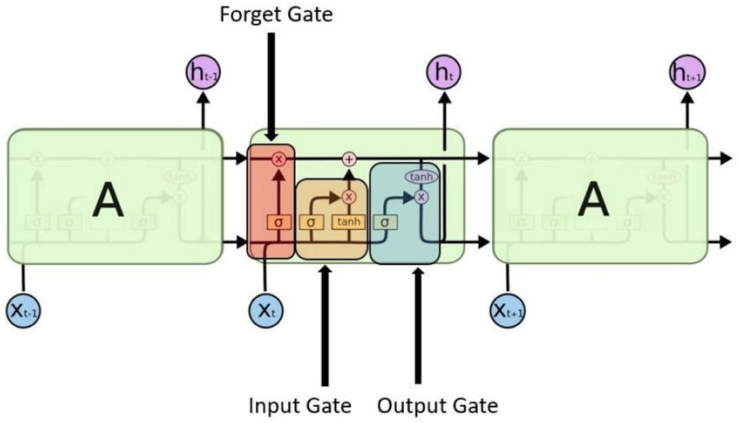


Figure 2.4: LSTM Gates (Rathor, 2018)

2.4.2 Support Vector Machines

Support Vector Machine is a discriminative classifier defined by a separating hyperplane i.e. given labeled training data the algorithm outputs an optimal hyperplane that categorizes new examples (Reddy, Sentiment Analysis using SVM, 2018). In two-dimensional space, this hyperplane is a line dividing a plane into two parts wherein each class lay on either side. Support vectors are data points that are closer to the hyperplane (points that are close to the opposing class) and influence the position and orientation of the hyperplane. Hyperplanes are decision boundaries that help to categorize data points. Data points falling on either side of the hyperplane might be likened to different classes (Patel, 2017). The SVM algorithm aims to establish a hyperplane that gives the largest margin to training samples. Points falling closer to the hyperplane are called support vectors.

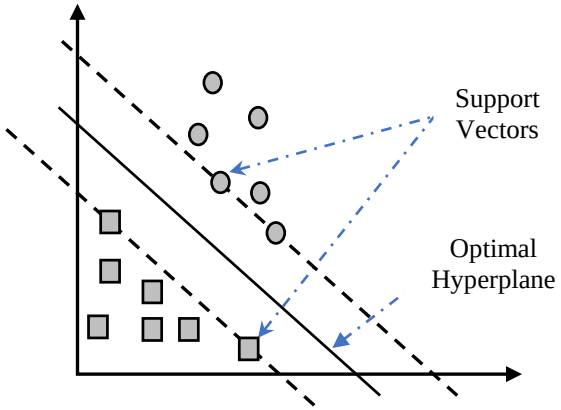


Figure 2.5: Optimal Hyperplane and Support Vectors

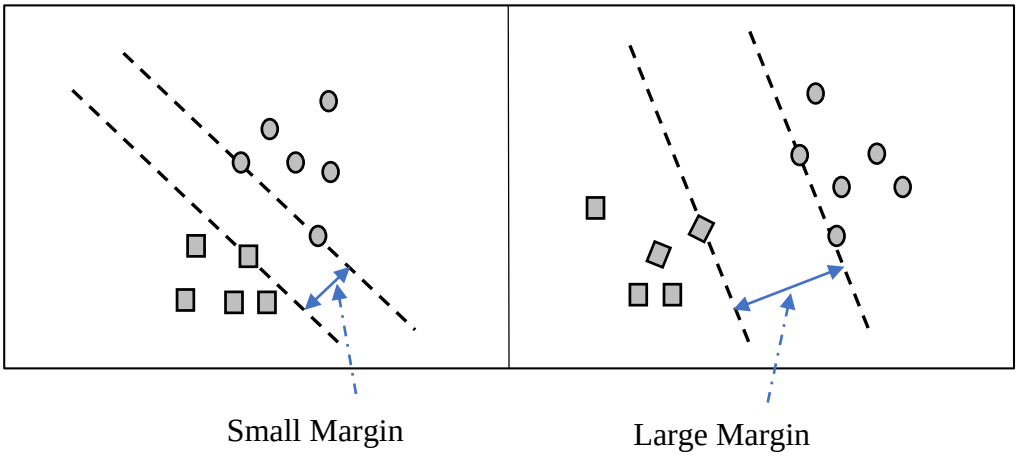


Figure 2.6: SVM Margins

Equation of the hyperplane is given by:

$$f(x) = w^T x + b \qquad \textit{Equation (2.1)}$$

At the hyperplane, the target value is equal to 0 (Liu, 2007). Therefore, the equations of the two planes at each of the support vectors is given by:

For a target value equal 1

$$w^T x + b = 1 \qquad \textit{Equation (2.2)}$$

and for a target value equal to -1,

$$w^T x + b = -1 \qquad \textit{Equation (2.3)}$$

SVM supports regression and classification tasks and can handle multiple continuous and categorical variables (Reddy, Sentiment Analysis using SVM, 2018). SVM uses an iterative training algorithm to construct an optimal hyperplane, which is used to minimize an error function (Hill & Lewicki, 2007).

2.4.3 Naive Bayes

This is a probabilistic machine learning model that's used for classification tasks and is based on the Bayes theorem (Gandhi, 2018). Below is the general Bayes equation.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \qquad \textit{Equation (2.4)}$$

The probability of *A* happening can be determined, given that *B* has occurred. *B* is the evidence and *x* is the hypothesis. The assumption made is that the features are independent of each other i.e. presence of one particular feature does not affect the other. Types of Naive Bayes Classifiers include Multinomial Naive Bayes, Bernoulli Naive Bayes, and Gaussian Naive Bayes (Gandhi, 2018). Based on this classifier a more general classifier known as Statistical n– Gram Modeling can be formed. This modeling technique is more accurate with regards to text classification. The technique considers dependence between adjacent words (Peng, 2003).

2.4.4 Text Classification Process

This is the process of classifying a text statement into a pre-defined class (Joachims, 1998). With regards to this research, the text statement is a tweet and the category is a value indicating if the tweet is radical or non-radical. The process occurs in the following steps: reading tweet datasets, feature extraction, and creation of a model. Figure 2.1 illustrates how these steps are applied when training a machine learning model while Figure 2.2 illustrates how a piece of text is classified when it is passed onto a developed text classification model.

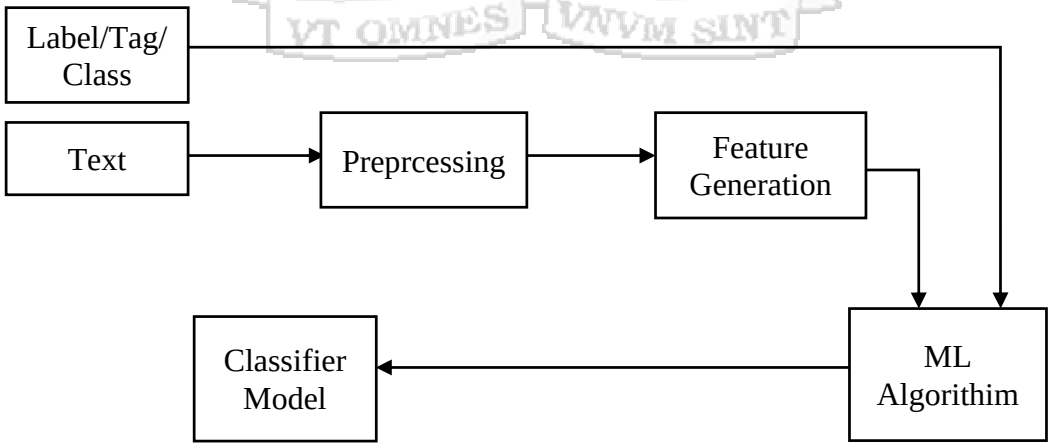


Figure 2.7: Text Classifier Development (Geitgey, 2018)

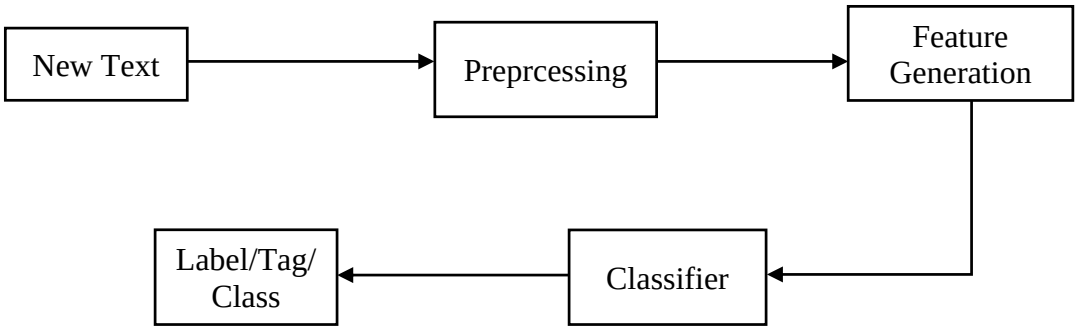


Figure 2.8: Text Classification Process

2.4.1.1 Pre-Processing

This is the process in which the data is transformed, or encoded, to get it to such a state that the computer can now parse it easily (Pandey, 2019). When handling this step, there are various steps/stages which the data goes through. In the related works under text classification techniques for detection of radicalization (Weber, Darwish, & Magdy, 2016); the preprocessing is summarized into two major steps i.e. *noise removal* and *tokenization*. For this research, an additional normalization step is added hence three steps were followed as illustrated in figure 2.3 below.

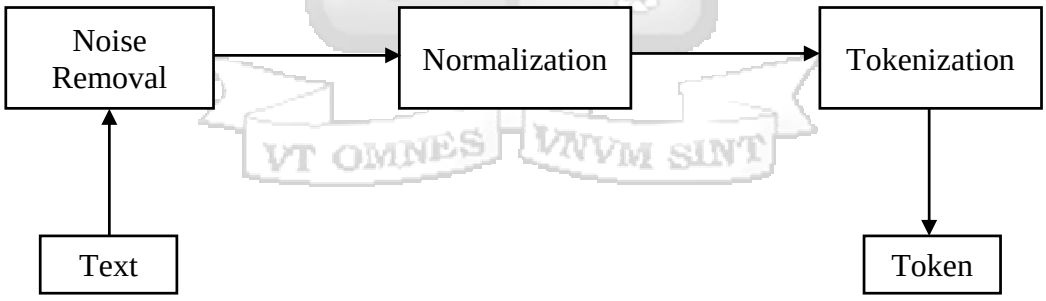


Figure 2.9: Pre-processing Stages

The raw tex data will first go through *Noise Removal* to ensure all unwanted characters and stopwords are removed. The output is then *Normalized* to ensure uniformity of all text instances such as having all text be in lower case form. The normalized text is then broken up into small units in a process referred to as *Tokenization*. By introducing normalization in pre-processing there is room for improving class prediction accuracy (Ganesan, 2019).

2.4.1.2 Feature Vectors and Document Representation

A feature vector is an n-dimensional vector of numerical features that represents some object (Eibe & Witten, 2005). Algorithms used in machine learning require a numerical representation of feature vectors as this facilitates mathematical computation and statistical analysis. Feature selection is the process of selecting a subset of the terms occurring in a document and using only this subset as features in text classification (Selamat & Zainuddin, 2014). Features may include word count, presence of words, presence of punctuation marks, and time-based features such as when a tweet was published. Feature selection methods fall under three categories (Gajare, 2019). The first is the filter method which uses chi-square or information gain techniques for selection. The second is the wrapper method that relies on heuristics using either Recursive Feature Elimination, Forward Selection, or Backward Selection for selecting features. The third is the embedded method that learns which features from the dataset and tries to produce features that best contribute to the accuracy of the classification model. For this research, the embedded model was preferred due to the requirement of learning feature vectors.

After feature selection, weights are applied to features using a chosen weighting scheme. There exist a number of weighting schemes (Reddy, Sentiment Analysis using SVM, 2018), these include: Word Embeddings; Word Frequencies with TF IDF Vectorizer; Bag of Words Model (Boolean Scheme) and Word Counts with Count Vectorizer.

Bag of Words (BoW)

A text is described in this model as the bag (multiset) of its words, disregarding grammar and even word order but retaining multiplicity hence the size of the BoW dimension will be directly proportional to the size of the corpus used (Singh P. , 2019). Once a vocabulary of known words from a corpus is formed, words within a sentence within the corpus are represented as either 1 or 0 depending on whether they are present in the corpus or not. Given a corpus of sentences: *'This is a good child' & 'This was a bad child*' the vocabulary would be: *'this, is, a, good, child, was, bad'* that can then be represented as seven numerals i.e. *'this:1, is:2, a:3, good:4, child:5, was:6, bad: 7'*. The feature representations of the tow sentences will then be as illustrated in Table 2:2 below:

Table 2.2 BoW Presentation

Corpus Vocabulary	*This*	*Is*	*A*	*Good*	*Child*	*Was*	*Bad*
	1	*2*	*3*	*4*	*5*	*6*	*7*
Sentence 1	*1*	*1*	*1*	*1*	*1*	*0*	*0*
Sentence 2	*1*	*0*	*1*	*0*	*1*	*1*	*1*

Given the 0 or 1 presentation used by BoW, every word in a document is assumed to be independent hence it does not allow for the presentation of similar words (Singh P. , 2019). Additionally, BoW results in an extremely sparse matrix, since there is a nonzero value in the dimensions corresponding to the terms in the sentence.

2.4.1.3 Word Embeddings

A word embedding converts a word to an n-dimensional vector (Cam-Stei, 2019). Words that are related such as '*desk*' and '*table*' map to similar n-dimensional vectors, while dissimilar words such as '*car*' and '*pen*' have dissimilar vectors. Through this representation, a word's meaning is reflected based on its embedding hence a machine learning model can learn how words in a given document relate to each other. The benefit of this method is that a model trained on the word '*desk*' will be able to react to the word '*table*' even if it had never seen that word in training. To tackle the high dimensional issues, word embeddings use pre-defined vector space such as *200* to present every word. The vector space dimension is fixed regardless of the size of the corpus. Most word embedding dimension varies between 8 dimensional (for small datasets), up to 1024 dimensions when working with large datasets. Higher-dimensional embeddings can capture detailed relationships between words but take more time to learn. The vector space then has each word as a vector represented with values within the defined dimension. To determine the vector representation values, two techniques can be adopted depending on factors such as time resource needed and accuracy (Kulshrestha, 2019) These are the Continuous Bag of Words (CBOW) approach and the Skip-gram approach.

When using the Continuous Bag of Words method, a neural network is used to inspect the immediately surrounding context words and predict a target word that comes in between. The Skip-Gram method, on the other hand, uses a neural network to inspect the target word and predict the immediately surrounding context words. For regular terms, CBOW works much faster and has a slightly better accuracy while Skip-gram works well with small quantities of data, and even with uncommon words or phrases (Chia, 2018).

2.4.1.4 Term Frequency Inverse Document Frequency

The TF IDF (Term Frequency Inverse Document Frequency) scheme is the most popularly used with regards to text classification tasks. It can be calculated by getting the product of the TF value and the IDF value. TF of a word denoted as *c* is the frequency of that word in a given document denoted as *d* (Bartosz, 2018).

$$TF = \frac{occurrences\ of\ word}{number\ of\ words\ in\ document}$$

IDF of a word c is the measure of its significance in a collection of documents which in this case would be the corpus used in constructing the model (Bartosz, 2018).

$$IDF = log\frac{total\ number\ of\ documents}{document\ with\ word}$$

The TF-IDF is what's referred to as the weight of a given word/term in a collection of documents. The weight of each term can then be computed using different weighted schemes namely Boolean value, Term Frequency (TF), Inverse Document Frequency (IDF), Term Frequency, and Inverse Document Frequency (TFIDF). TFIDF scheme is the most widely used when computing the weight value of each term. Given a document denoted as doc $_i$, the weights of words in the document can be described as $W_{i1}, W_{i2}, W_{i3}, \ldots W_{ij}, \ldots\ldots W_n$ where W_{ij} is the is TFIDF value of j th term in an n-dimensional vector space. A statistical language model is a probability distribution over sequences of words (Singh, Devi, & Anjana, 2017). This technique is mostly used in application areas such as speech recognition, machine translation, document classification, optical character recognition, information retrieval, and handwriting recognition. The most commonly used technique in the statistical language model is the n-gram language model. The n-gram language model assumes that the probability of a word appearing in a document depends on the previous n words (Singh, Devi, & Anjana, 2017). The probability $P(W_1, W_2, W_3, \ldots\ldots W_m)$ of observing the sentence $W_1, W_2, W_3, \ldots\ldots W_m$ is approximated as:

$$P(W_1, W_2, \ldots\ldots W_m) = \prod_{i=1}^{m} (W_1 | W_1, \ldots\ldots\ldots\ldots W_{i-1})$$

2.5 Related Work

There has been a research work done concerning sentiment analysis and text classification using Twitter as a source of data. This section reviews additional research projects done concerning sentiment analysis using machine learning algorithms.

2.5.1 Sentiment Analysis Using Polarity Dictionaries

In their work on the use of polarity dictionaries (Agarwal, Xie, Vovsha, Rambow, & Passonneau, 2011), they proposed an approach where tweets were classified as being negative, neutral or positive using features were based on the polarity of words. This was determined by using dictionaries like WordNet which assigns each word a score between 1 and 3. Other features include feature count and presence of exclamation marks and capitalized text. The polarity of words features, counting, and presence of exclamation marks and capitalized text formed what was called senti-features. In the experiments using an SVM classifier and unigram features, the researcher got 71.36% accuracy. In a second approach, unigram features were combined with senti-features and the accuracy result increased to 75.39% showing the contribution of the senti-features for tweets sentiment classification.

2.5.2 Twitter Sentiment Classification using N-grams

Bigrams were used to classify tweets that contain negated phrases like “not good” or “not bad” (Go, Bhayani, & Huang, 2009). An n-gram is a language modeling technique whereby a sequence of words is used to predict the next term in the sequence. For this study, the researchers used bigram i.e. a sequence of two adjacent terms. For machine learning algorithms for text classification, they used Naive Bayes, maximum entropy, and SVM for text classification. Emoticons were stripped out from the training data since they introduced a negative impact on the accuracy of the maximum entropy and SVM classifiers. This approach allowed the classifiers to learn from other relevant features they use like unigrams, bigrams, or part of speech.

2.5.3 Twitter Trending Topic Classification

Twitter trending topics were classified into 18 different categories like sports, politics, technology, etc. A bag of words approach for text classification was used whereby for each topic, a document was made from trend definition and a varying number of tweets. The tf−idf weights were computed for each word. The tf−idf measure allowed evaluating the importance of a word to a document. This tf−idf was used to filter out common words. For each of the 18 labels, topmost 500 or 1000 frequent words with their tf−idf weights were used

to build the dataset for model development. The best accuracy was obtained from using the Naive Bayes Multinomial classifier (65.36%) which performed better than the native Naive Bayes (45.31%) and SVM (61.76%) classifiers (Lee, Palestia, Narayanan, Agrawal, & Choudhary, 2011).

2.5.4 Real-time sentiment analysis for the detection of terrorist activities in Kenya

A model was developed to establish crime patterns associated with terrorist activities using sentiment information deduced from twitter data. Tweets were collected in a span of seven days and stores in an SQL database. A dictionary used for classification was developed to train the model alongside collected data using both Bayes and Support Vector Machine algorithms. This approach however required a predefined classification scheme i.e. use of a dictionary hence score would be impacted by missing adequate training data. The final developed model performed classification using an ordinal scale and achieved an accuracy score of 73% (Ngoge, 2016).

2.6 Proposed Approach

The methods identified for use in detecting radicalization rely heavily on the use of human intervention as a final decision making element. The Link Based Bootstrapping, for instance, needs human personnel to manually download and look into data from suspected radicalization websites URLs. The alternative text classification method also relly on the use of the Bag of Word feature representation approach which as discussed sufferers various limitations. Finally, all of the identified locally used techniques in identifying radical groups are too manual needing close human supervision/guidance.

I proposed an automated text classification approach that aims to provide a solution to the limitations of the above-identified techniques. Our approach used the embedding feature representation technique to generate machine learning features that I then used to build an artificial neural network classifier for sentiment analysis.

2.7 Conceptual Framework

Below is a conceptual framework for the research. The framework is based on the literature review and the identified gaps/opportunities. Relevant datasets in the form of tweets will be collected from open source data repositories stored. The data was then be pre-processed and used to generate needed embeddings as well as to train the text classification model applicable in use for detecting radicalization.

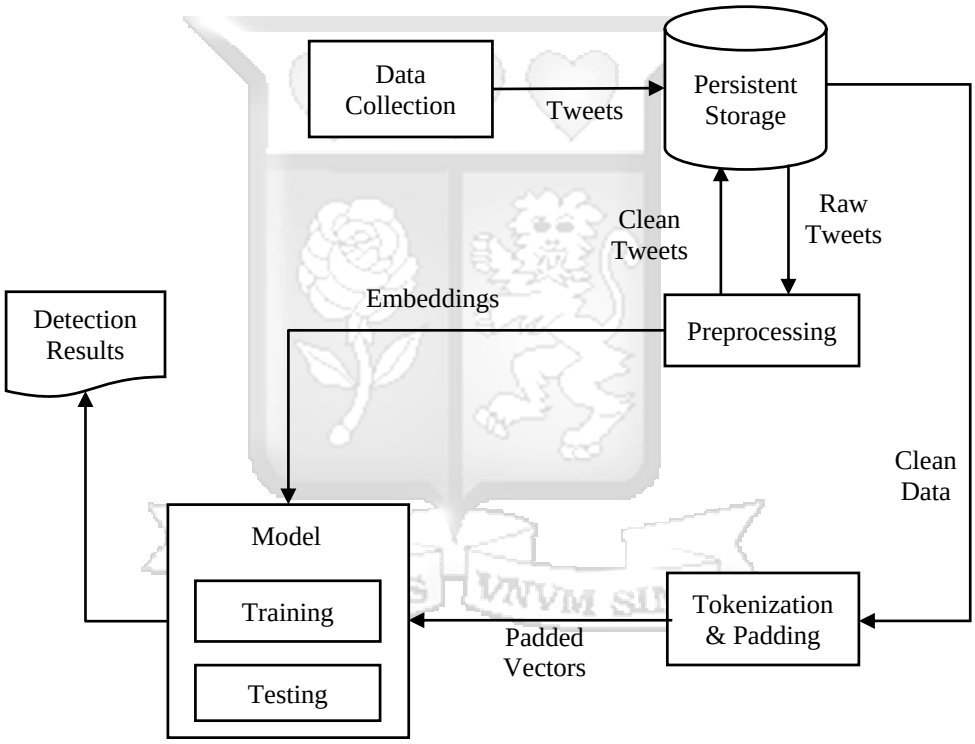


Figure 2.10: Conceptual Framework

Chapter 3 Research Methodology

3.1 Introduction

This section describes the various methods and procedures that will be used in carrying out this particular research. This research will be guided by the objectives set in chapter one. This research will borrow knowledge from previous related approaches as well as borrow the knowledge of the techniques used in those previous approaches. Historical tweet posts that have been classified as positive for relaying radicalized communication, as well as newly collected tweets, will be used as primary data for training the model. A number of experiments were designed to validate the model and determine the approach to adopt.

3.2 Research Design

Research design is the structure that the research process follows in terms of data collection, analysis of data, and evaluation of results to answer research questions and achieve research objectives**.** This research will follow an experimental design approach involved the identification of research objectives, the building of sentiment analysis model, validation of the model using a number of experiments to ensure the best performance (Creswell, 2014)**.**

3.3 Datasets and Data Collection

To build an adequate dataset of various tweets, this research relied on data obtained from two sources that contained Twitter data available to the public for research and analysis purposes. The reason for using second-hand data was mainly to save on resources i.e. time since the collection of similarly sized first-hand data would take longer and, secondly the data obtained had enough instances needed to create a large dictionary set that was vital when creating word embeddings. This set of data was used in forming the corpus used in implanting this research's proposed solution. Additionally, a third dataset was also obtained to facilitate two of the validation experiments. The third set was location-specific to have relevant word representation for a given location as people tend to use words differently in various regions; this helped to show that how a model can be customized to classify tweets in a given region/location just by changing the input to its embedding layer during training. This additional dataset was thus explicitly mined from twitter giving into consideration the location geotags and other boundary defining parameters that could assist in defining the boundaries of which to mine the data. The data sources were:

3.3.1 Kaggle Data Repository

The dataset of interest from this repository is titled *How ISIS Uses Twitter*. It contains over 17,000 tweets from 100+ pro-ISIS fanboys from all over the world since the November 2015 Paris Attacks (How ISIS Uses Twitter, 2016). Some of the attributes of interest in the dataset were: Username, Location, Date, and timestamp of the tweet and The tweet itself

3.3.2 Sentiment 140 Project

This was a project by Stanford University aimed at improving analysis in the field of natural language processing (Go, Bhayani, & Huang, 2009). The dataset has 1,600,000 tweets extracted using the Twitter API. The tweets have been labeled as 0 for negative, 2 for neutral, and 4 for positive. Of interest to this research was the positively labeled tweets to help in creating positive instances of the training corpus used for model construction. The assumption made here was: a tweet exhibiting a positive sentiment is unlikely to be radical thus would be different from a radical tweet is not just in terms of the sentiment but also in terms of choice of words used in the tweet. Around 17,000 instances labeled 4 were chosen during corpus construction. These instances were later labeled as 1 to depict instances of non-radical posts on the social networking platform.

3.3.3 Mining Data from Twitter

To create the relevant datasets for developing adequate word embeddings, this research relied on Twitter API to assist with data mining. Twitter has two APIs that can be used by developers to collect tweets using: The Streaming API and the Representational State Transfer (REST) API. Both APIs require the use of Open Authentication (OAuth) to allow applications to get access to them and issue responses in JavaScript Object Notation (JSON) format.

Advantages of Streaming API:

i. The Streaming API allows developers to process tweets in real-time, continuously delivering responses in JSON format over long-lived HTTP connections (Twitter, 2019).

Advantages of REST API:

i. The REST API enables developers to read and write Twitter data.
ii. The REST API enables developers to query against indices of recent tweets up to 7 days old.

To interface with the API, a python library, Tweepy was used to allow for specifications of various parameters to aid in the collection of tweets during the development and

implementation of the model. This library allows for the use of a variety of search parameters as depicted in the table below to improve relevance during data mining on Twitter.

3.4 Dataset Pre-processing

Data obtained through twitter usually contains a lot of unuseful information in its raw form. This is not suitable for machine learning as it can have adverse effects on a model's ability to classify instances accurately. Moreover, it can alter the accuracy of results, and therefore preprocessing the dataset is necessary. To solve for this the following preprocessing steps described in Chapter two were carried out: These were: Noise Removal, Normalization, and Tokenization. Given the various information one can consider as noise, I opted to only remove the following elements during the noise removal step:

i. Remove Retweet tag and Username.
ii. Remove all URLs links.
iii. Removal of HTML character codes rendered during the collection of tweets.
iv. Removal of Stop-Words. This is generally the most common words in a language such as “the”, “is”, “a” etc.

After preprocessing, the results were stored in a comma-separated values (csv) files with a column for the tweets in their raw form as well as a column for the final tokens generated from each tweet.

3.5 System Development Methodology

The prototype this research proposed was developed using the Rapid Application Development (RAD) system development methodology. RAD is a lifecycle used for the development of software which provides faster development and also gives high-quality software than by using traditional software development lifecycle. RAD facilitates organizations in the development of software faster and it also helps reducing development cost and maintains the quality of software (Naz & Khan, 2015).

3.5.1 Phases of RAD

Below is a diagram and description of the four phases of this methodology:

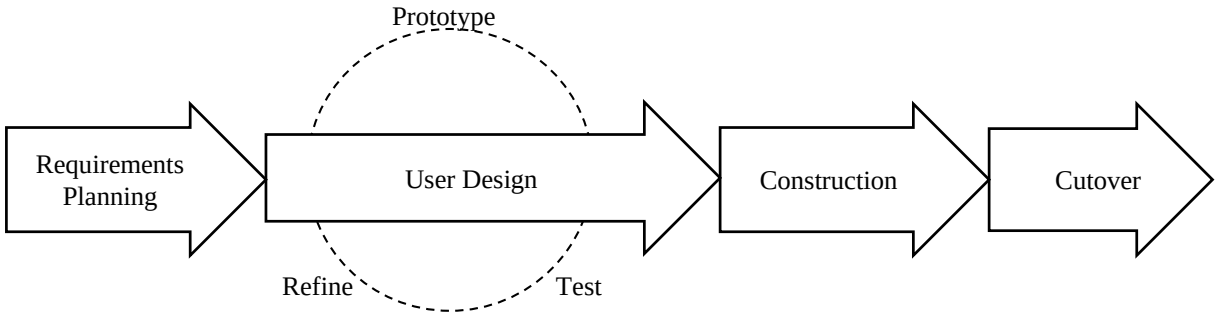


Figure 3.1: Phases of RAD

3.5.1.1 Requirements planning

With regards to this research this phase involved:

i. The definition of the research problem.
ii. Understanding existing solutions using similar techniques.
iii. Identifying gaps in existing solutions.
iv. The definition of requirements of the proposed sentiment analysis model.

3.5.1.2 User Design

This involved the development of a blueprint of the proposed prototype using Unified Modeling Diagrams (UML) to illustrate the various components of the proposed prototype.

3.5.1.3 Rapid Construction

Based on the designs from the design phase the model was developed using Python programing language. Pandas Library for Python will provide data structures to help in the analysis of the datasets. The Tensorflow-Library which is a free software machine learning library for the Python will be used to implement various machine learning algorithms.

3.5.1.4 Cutover/Transition

After successful construction and approval from relevant bodies/institutions, the model can be implemented to help detect radicalization on the social media platform, Twitter.

3.5.2 Reasons for choosing RAD

The methodology was chosen given it is convenient as it allows for quick development of prototypes at high quality and fewer costs. RAD also allows for the reusability of components due to modularity and prototyping which allowed for multiple iterations during development to ensure the final iteration had achieved its highest possible accuracy.

3.6 Research Quality

The quality of this research was based on the accuracy of the model in classifying labeled tweets. A testing set that contains labeled tweets was used to evaluate the model and produce results for the confusion matrix. The set of four comparison categories within the confusion matrix was used to calculate various performance parameters. These categories were:

i. True Positives (TP) – Correctly classified positive tweets.
ii. True Negatives (TN) – Correctly classified negative tweets.
iii. False Positives (FP) – Incorrectly classified positive tweets.
iv. False Negatives (FN) – Incorrectly classified negative tweets.

The accuracy of the model was then determined using the formula below:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \qquad \textit{Equation (3.1)}$$

The precision of the model i.e. the ratio of correctly classified tweets to the total number of tweets classified under a particular category (Feldman & Sanger, 2006) is given by:

$$Precision = \frac{TP}{TP+FP} \qquad \textit{Equation (3.2)}$$

The recall of the model i.e. the number of correctly classified tweets among all tweets in belonging to a given category (Feldman & Sanger, 2006) is given by:

$$Recall = \frac{TP}{TP+FN} \qquad \textit{Equation (3.3)}$$

The F-Score i.e. the weighted harmonic average of the test's Precision and Recall (Feldman & Sanger, 2006), is given by:

$$F - Score = \frac{Precision\ x\ Recall}{Precision+Reacll}\ x\ 2 \qquad \textit{Equation (3.4)}$$

3.7 Performance Visualisation

Tables and graphical representations were used to illustrate the model performance. Tables were used to display the accuracy, precision, recall, F- score values of the model and to displays test results when comparing with other alternative models. Graphs were used to plot the Accuracy and Loss values during different pass-throughs of the training ad testing data during model development. The graphs helped illustrate the improvement of the model an also its peak point where no more training ant testing was required as it would not benefit further.

3.8 Model Validation

To validate the approach taken by this research, two experiments were carried out to provide a basis for comparison of performance metrics.

3.8.1 Experiment 1: Using Alternative Machine Learning Algorithms

This experiment aimed to compare the performance of the LSTM model proposed by the researcher with other machine learning algorithms i.e. Naïve Bayes and Support Vector Machine using the similar dataset for both training and testing

3.8.2 Experiment 2: Using Different Word Vocabularies

This experiment aimed at testing how different vocabularies trained/developed by third-party datasets can be used in the prediction model and how they would affect the accuracy scores. This would also demonstrate the ability to customize the model to specific locations without having to retrain the entire model from scratch.

3.9 Ethical Considerations

This research aims to utilized publicly available online data from social networking platforms, Twitter. The data will be collected and modified only by the act of pre-processing to allow for easier study and analysis. Additionally, the research was granted an ethical approval certificate by a research ethics review committee and a research permit from the National Commission for Science, Technology, and Innovation, Kenya.

Chapter 4 Design and Architecture

4.1 Introduction

This chapter describes the overall architecture and detailed design of the proposed solution by incorporating various requirements. UML diagrams will be used to: describe the general architecture of the solution, provide a detailed description of each component of the solution, and to illustrate user interaction procedures with the various components.

4.2 Design

Given the main objective of this research is to develop a model for detecting radicalization on Twitter; this section outlines the various requirements needed by the proposed solution.

4.3 Functional Requirements

The functional requirements are:

i. The prototype should allow a user to enter keywords to be used as search parameters in the retrieval of twitter data to be classified. These include User Account names, Location names/coordinates, and Topic Hashtags. For example to classify tweets in a given location, one would have to specify this to the system.

ii. The prototype system should retrieve tweets from Twitter using the Twitter Search API matching the keywords specified by the user.

iii. The prototype system should preprocess the retrieved tweets to have them ready for classification.

iv. The prototype system should then feed the tweets to them a model for classification. The model also handles text tokenization and text padding

4.4 Non-Functional Requirements

4.4.1 Persistent Storage

The porotype solution should provide permanent storage for tweets both used as training and testing data as well as user queries. Such data can be used as evidence in court cases or for tracking radicalization activities hence should also be easily retrievable at a moment's notice.

4.4.2 Data Integrity

The prototype should in no way alter the integrity of the messages within tweets during the various data cleaning processes such as stemming.

4.4.3 Scalability

The prototype should allow for dynamic scalability based on the amount of data available i.e. size of training and testing corpus. This should not adversely hinder the performance of the prototype in terms of the accuracy of the results produced.

4.5 Architecture

The system architecture illustrated in Figure 4.1 shows the general layout of the twitter radicalization detection prototype system which is based on the finding of this research. The classification process begins when a user inputs a user account handle to guide in retrieval relevant tweets from a user of interest. Apart from providing a username as an input, one can provide other accepted retrieval parameters such as locations and topic tags (hashtags) to widen the scope of which to detect instances of radicalization. The tweet collection module then streams and saves the relevant tweets as per the provided parameters. The tweets undergo pre-processing and are then stored into a csv file for use later in a *Pandas* data frame. Once imported into a data frame, the tweets are then transformed into vectors and used for the detection process. The transformed representation is then fed as an input to the classifier and hence a tweet is classified by being awarded a score. The score between 0 and 1 and as such is rounded off to be either 0 or 1 to imply a corresponding class labeled for each value. The class label is then presented back to the user.

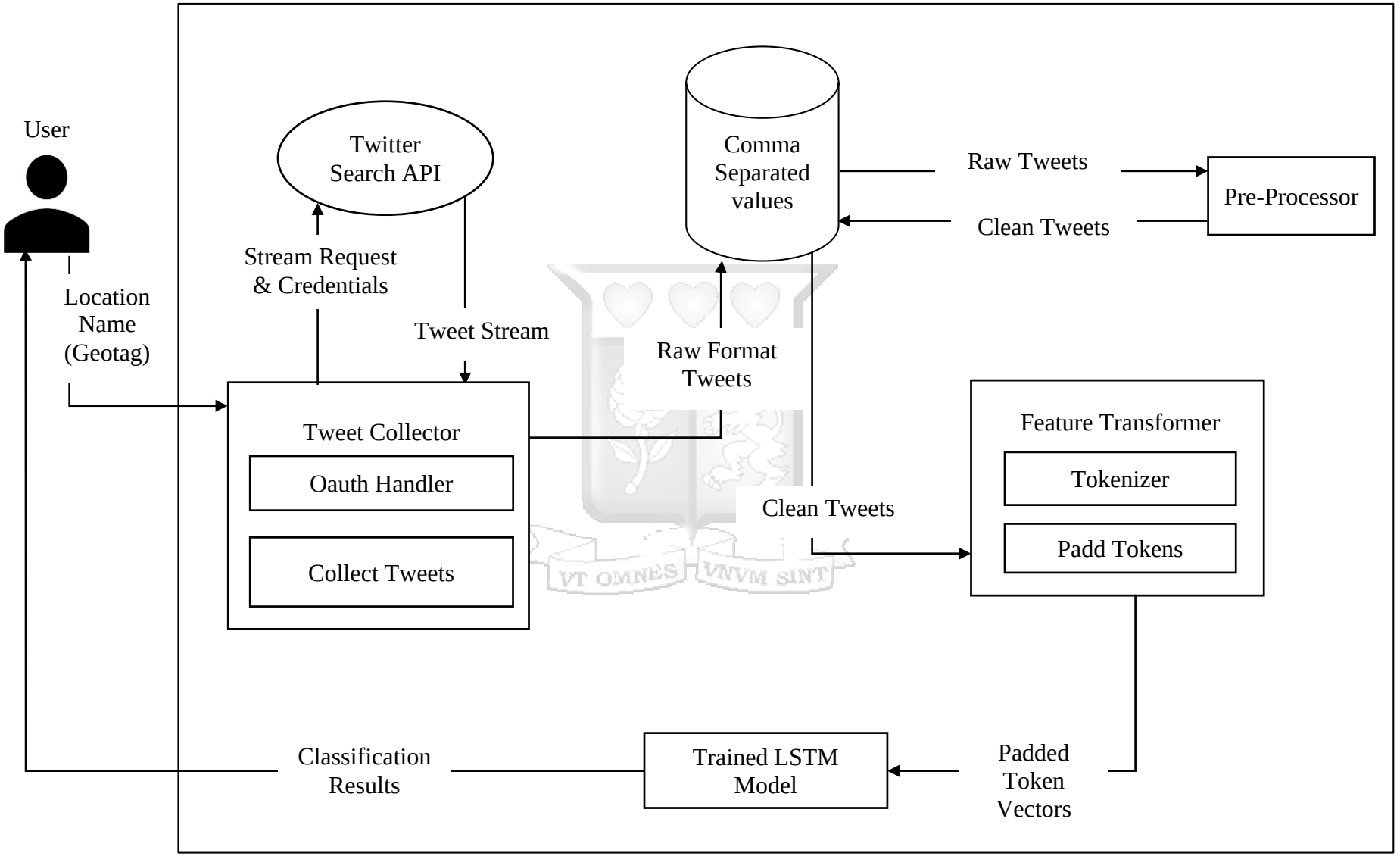


Figure 4.1: System Architecture

4.6 Use Case Diagram

Use case diagrams are used to illustrate the interaction between various actors and the prototype system. Figure 4.2 illustrates these interactions between the various actors and the proposed radicalization detection prototype.

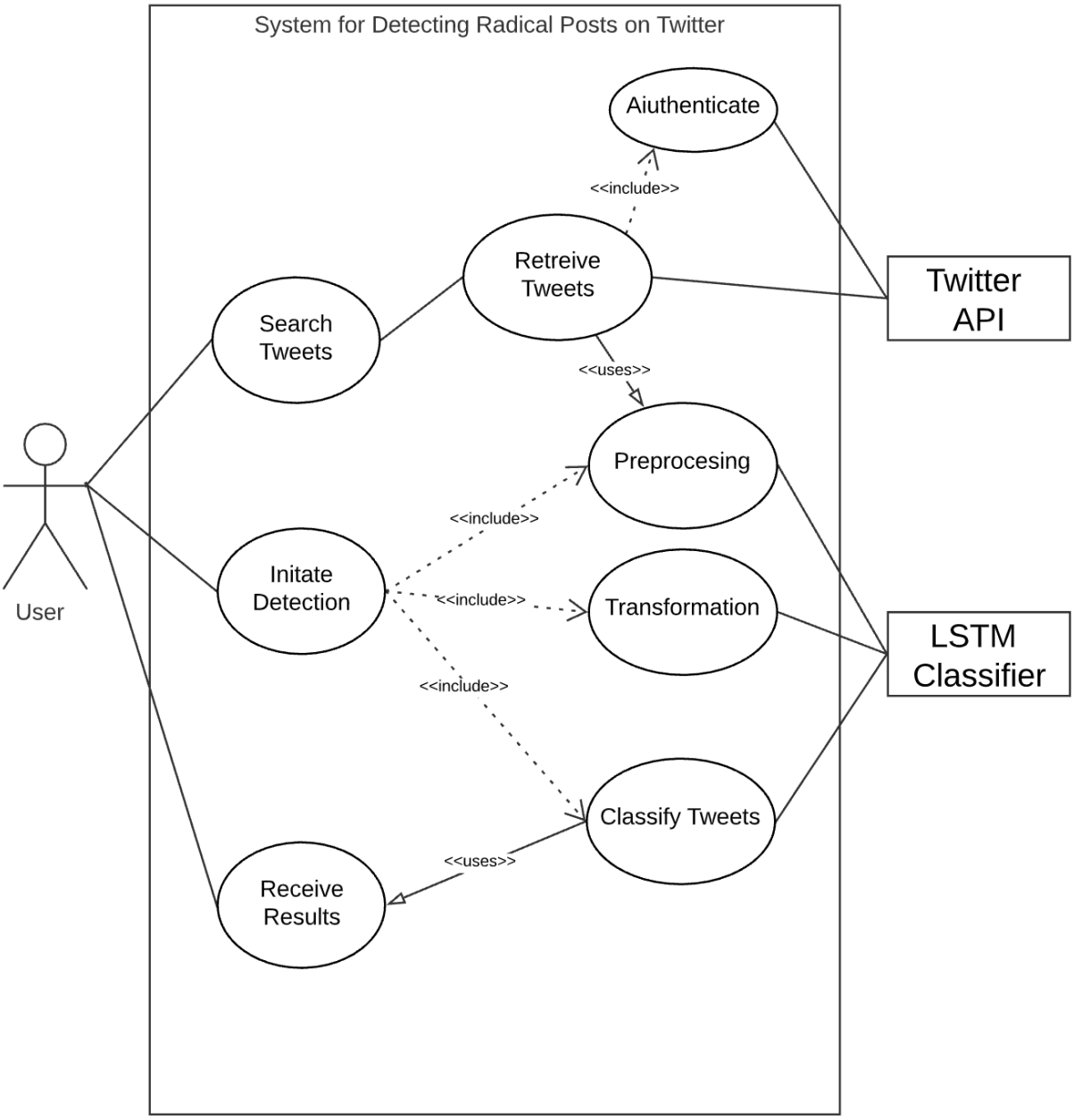


Figure 4.2: Use Case Diagram

4.6.1 Use Case Diagram Description

This section provides comprehensive descriptions for the use cases in Figure 4.2 in a two-column fully dressed format.

Use case: Search.

Primary Actors: User, Twitter API

Pre-conditions: User has access to the internet on platform/device being used.

Postconditions: Search parameters are generated and passed onto Retriev Use Case.

Use case: Retrieve Tweets

Primary Actors: Twitter API.

Pre-conditions: Search use case completed successfully.

Postconditions: Tweets are fetched from twitter based on search parameters.

Use case: Initiate Detection

Primary Actors: User, System

Pre-conditions: Retrieve use case completed successfully.

Use case: Pre-Process

Primary Actors: User, System.

Pre-conditions: Retrieve use case completed successfully.

Postconditions: Unwanted data(noise) is removed from tweets.

Use case: Classify Tweets

Primary Actors: System.

Pre-conditions: Transform use case completed successfully.

Postconditions: A test case is given a label based on predicted class.

Use case: Receive Class

Primary Actors: User.

Pre-conditions: Classify Tweets use case completed successfully.

Postconditions: User view the class i.e. Radical or Non-Radical.

4.7 Sequence Diagram

The sequence diagram depicted in Figure 4.3 below shows the sequence of various interactions between the user and the prototype as well as various interactions of internal components within the system. The user provides an account handle, keywords, location(geotag), or topic tag(hashtag) to facilitate data collation and thus form a corpus which is later fed into the detection prototype; the provided input is used as a parameter for the ***search ()*** function. Collected tweets are then saved into a csv file. The user then initiates the detection process. The collected tweets then read into a data frame for pre-processing using the ***load ()*** and ***clean_data ()*** functions. The output is then fed into the ***transform ()*** function to produce a numerical representation of the documents(tweets). The numeric representation is then fed into the ***classifier ()*** function which returns a class i.e. radical or non-radical. The ***clean_data (), transform (),*** and ***classify ()*** functions are repeated for all instances of collected tweets. The user then views the detection results using the ***view_result()*** function.

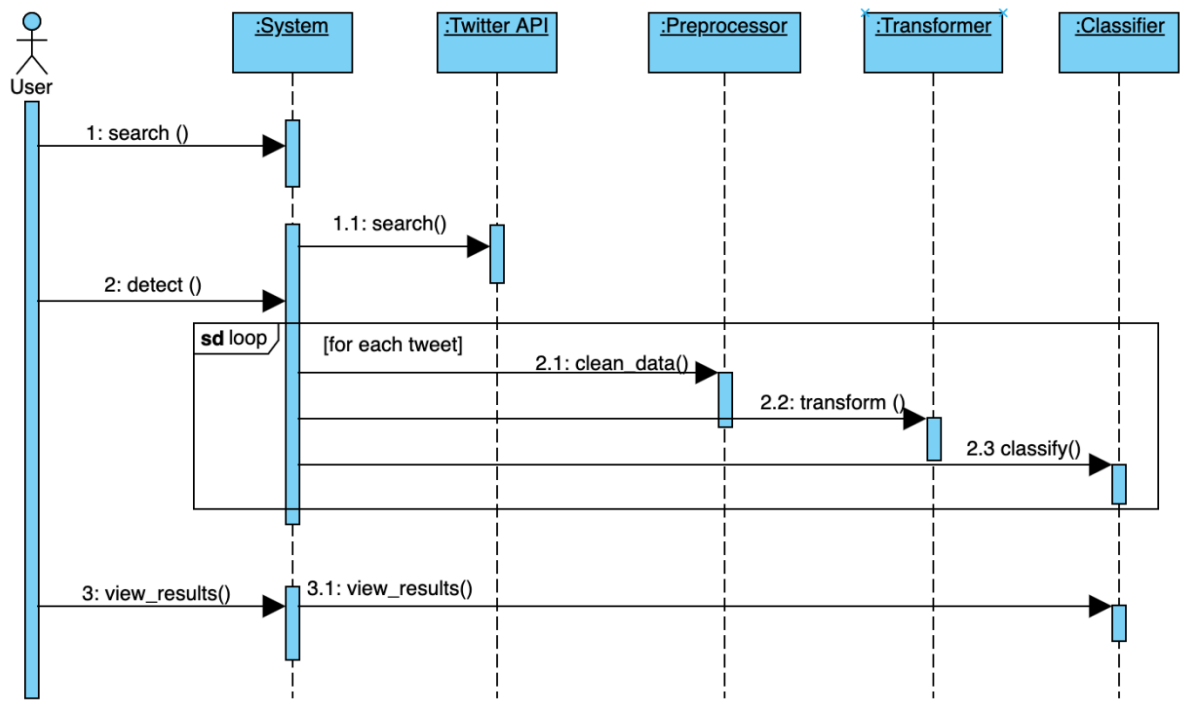


Figure 4.3: Sequence Diagram

4.8 Context Diagram

The context diagram as depicted in Figure 4.4 illustrates the various inputs and outputs from the prototype to the entities. The main entities interacting with the proposed prototype is the user(s). The user(s) issues a prediction request to the model which classifies the tweets and returns the labels of the tweets to the user. The second main entity is the Twitter API which facilitates the streaming of tweets from the online platform.

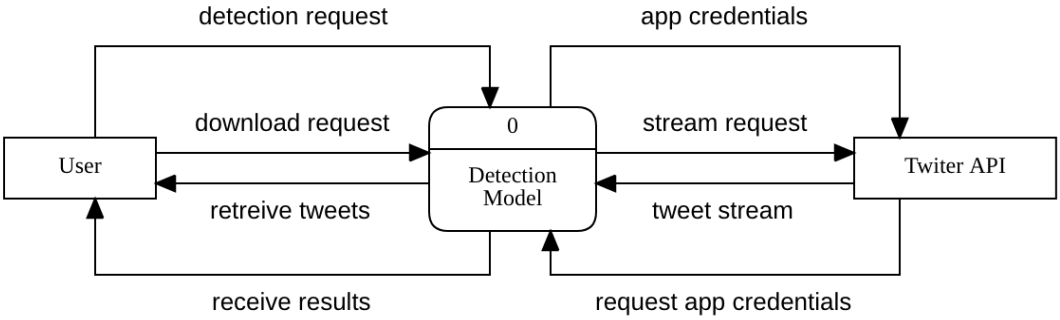


Figure 4.4: Context Diagram

4.9 Level 0 Data Flow Diagram

The level 0 DFD in Figure 4.5 below provides a more detailed view of the prototype's processes. The arrows indicate the flow of data between various components of the DFD. **Process 1: Collect Data** receives a stream request alongside relevant parameters from the user and passes the request to the Twitter Search API. The API validates the relevant credentials and streams the tweets to the user. These are then stored into a csv file in data store **D1**.

Process 2: Clean Data receives a detection request from a user, reads the csv file from data store D1 into a data frame and cleans them, and then stores them as cleaned data in data store **D2**.

Process 3: Transform reads cleaned data from data store D2 and transforms them into a numerical format that is suitable for machine learning.

The numeric representation is then fed into **Process 4**: **Classification** which labels the tweets as per the predicted nature of the data by the machine learning classifier.

Process 5: Results provide feedback to the user i.e. classification results.

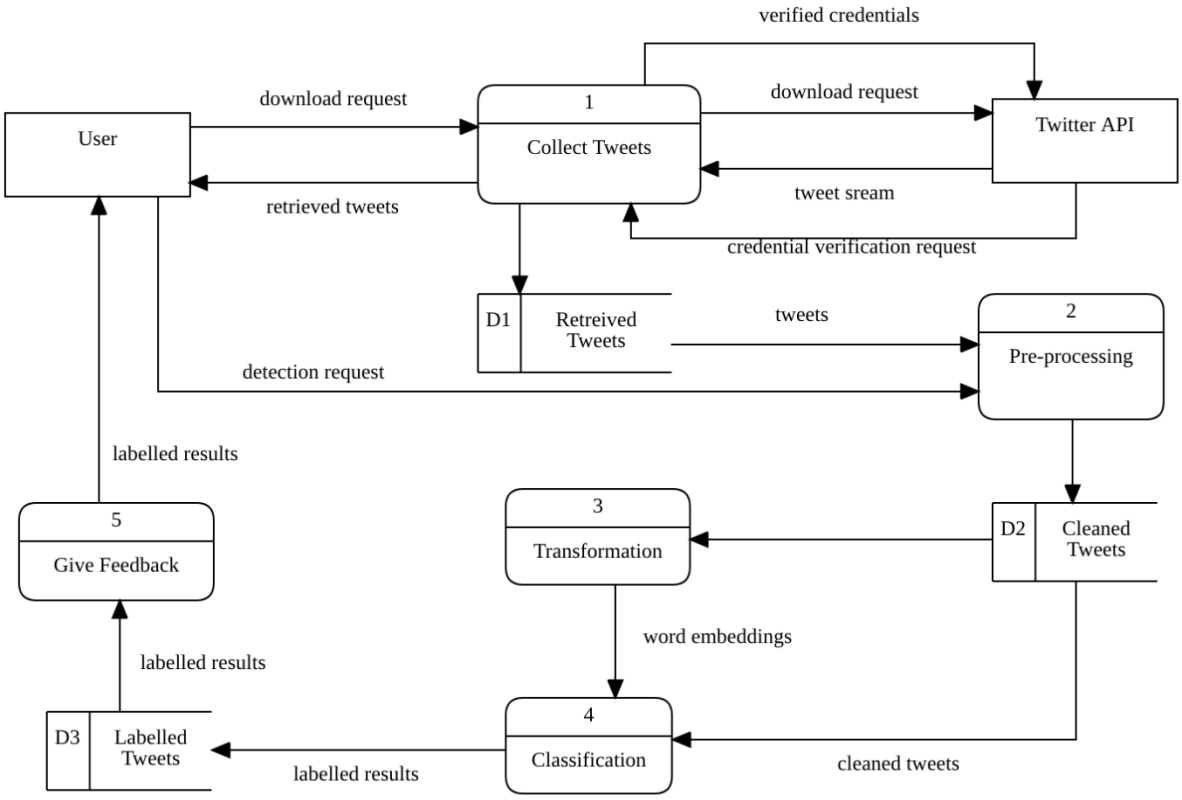


Figure 4.5: Level 0 Data Flow Diagram

Chapter 5 Implementation and Testing

This section describes how the prototype was implemented, tested, and validated. The first part describes basic exploratory data analysis procedures carried out to get more insight into the data. The second part covers the development of a relevant and adequate corpus for training and validating the proposed machine learning model. The third section then describes the various processes used in building the discussed model as well as performing validation of the model.

5.1 Corpus Creation

5.1.1 Data Collection

Data needed for this research were grouped into two categories i.e.

i. Tweets viewed to express a radical view.
ii. Tweets viewed as to be of positive sentiment.

5.1.2 Data Pre-processing

The data contained a lot of irrelevant information that is not needed for the text classification process. Figures 5.1 and 5.2 show a snippet of how the data in each category appear in their raw format.

name	username	tweetid	time	tweets
GunsandCoffee	GunsandCoffee70	2	1/6/2015 21:07	ENGLISH TRANSLATION: 'A MESSAGE TO THE TRUTHFUL IN SYRIA - SHEIKH ABU MUHAMMED AL MAQDISI: h
GunsandCoffee	GunsandCoffee70	3	1/6/2015 21:27	ENGLISH TRANSLATION: SHEIKH FATIH AL JAWLANI 'FOR THE PEOPLE OF INTEGRITY, SACRIFICE IS EASY' htt
GunsandCoffee	GunsandCoffee70	4	1/6/2015 21:29	ENGLISH TRANSLATION: FIRST AUDIO MEETING WITH SHEIKH FATIH AL JAWLANI (HA): https://justpaste.it/Meet
GunsandCoffee	GunsandCoffee70	5	1/6/2015 21:37	ENGLISH TRANSLATION: SHEIKH NASIR AL WUHAYSHI (HA), LEADER OF AQAP: 'THE PROMISE OF VICTORY': h
GunsandCoffee	GunsandCoffee70	6	1/6/2015 21:45	ENGLISH TRANSLATION: AQAP: 'RESPONSE TO SHEIKH BAGHDADIS STATEMENT 'ALTHOUGH THE DISBELIEV
GunsandCoffee	GunsandCoffee70	7	1/6/2015 21:51	THE SECOND CLIP IN A DA'WAH SERIES BY A SOLDIER OF JN: Video Link :https://www.youtube.com/watch?v=U
GunsandCoffee	GunsandCoffee70	8	1/6/2015 22:04	ENGLISH TRANSCRIPT : OH MURABIT! : https://justpaste.it/OhMurabit https://twitter.com/account/suspended
GunsandCoffee	GunsandCoffee70	9	1/6/2015 22:06	ENGLISH TRANSLATION: 'A COLLECTION OF THE WORDS OF THE U'LAMA REGARDING DAWLAH': https://justp
GunsandCoffee	GunsandCoffee70	10	1/6/2015 22:17	Aslm Please share our new account after the previous one was suspended. @KhalidMaghrebi @seifulmaslul123 @Cl
GunsandCoffee	GunsandCoffee70	11	1/10/2015 0:05	ENGLISH TRANSLATION: AQAP STATEMENT REGARDING THE BLESSED RAID IN FRANCE:https://justpaste.it/AC

Figure 5.1: ISIS Fan-Boy Tweets Data

The ISIS Fan-Boy Tweets dataset contained four columns i.e. Name, Username, TweetId, Time, and Tweets (How ISIS Uses Twitter, 2015). The main column of interest was ‘Tweets’ which contained posting from individuals in support of the ISIS group. This was used as the radical statements hence denoted to be radical/extremist or to be in support of radical ideologies. The total number of instances was about 17,000.

4	3	Mon May 11 03:17	kindle2	tpryan	@stellargirl I looooooooovvvvveee my Kindle2. Not that the DX is cool, but the 2 is fan
4	4	Mon May 11 03:18	kindle2	vcu451	Reading my kindle2... Love it... Lee childs is good read.
4	5	Mon May 11 03:18	kindle2	chadfu	Ok, first assesment of the #kindle2 ...it fucking rocks!!!
4	6	Mon May 11 03:19	kindle2	SIX15	@kenburbary You'll love your Kindle2. I've had mine for a few months and never looked bac
4	7	Mon May 11 03:21	kindle2	yamarama	@mikefish Fair enough. But i have the Kindle2 and I think it's perfect :)
4	8	Mon May 11 03:22	kindle2	GeorgeVHulr	@richardebaker no. it is too big. I'm quite happy with the Kindle2.
0	9	Mon May 11 03:22	aig	Seth937	Fuck this economy. I hate aig and their non loan given asses.
4	10	Mon May 11 03:26	jquery	dcostalis	Jquery is my new best friend.
4	11	Mon May 11 03:27	twitter	PJ_King	Loves twitter

Figure 5.2: Sentiment 140 *Data*

This research sought the use of the Sentiment 140 dataset which is automatically labeled based on positive words, emoticons, and context of the tweet (Sentiment 140, kein Datum). The dataset had no pre-set column headers hence headers were assigned according to the description of each column i.e.

i. Column 1 – Sentiment Label
ii. Column 2 – Index
iii. Column 3 – Time
iv. Column 4 – Account Name
v. Column 5 – User Account Handle
vi. Column 6 – Tweets

The irrelevant extra information within the data can alter the accuracy of results and therefore pre-processing the dataset is necessary for all collected/acquired data hence the data were subjected to the three preprocessing stages described in chapter three i.e. Noise Removal, Normalization, and Tokenization.

To carry out Noise Removal and Normalization, a function was written to perform the two tasks. As much as all dataset depicted similarities in terms of structure and attributes, the function had to either have custom code added/ commented out to ensure the final output from all datasets was the same in terms of attributes. The code snippet in Figure 5.3 depicts a general form of the function used to achieve data cleaning.

```
def cleaner2():
    df.tweets=df.tweets.astype(str)
    df['Tweet'] = np.vectorize(cleaner1)(df['tweets'], '@"[\w]*')
    df['Tweet'] = df['Tweet'].str.replace("[^a-zA-Z#]", " ")
    df['Tweet'] = df['Tweet'].str.replace(r'\b(\w{1,2})\b', '')
    df['Tweet'] = df['Tweet'].str.replace("ENGLISH TRANSLATION ", " ", False)
    df['Tweet'] = df['Tweet'].apply(lambda x: x.lower())
    df['Tweet'] = df['Tweet'].str.replace("b RT :", "", False)
    df['Tweet'] = df['Tweet'].str.replace("https", "", False)
    df['Tweet'] = df['Tweet'].str.replace("http", "", False)
    df['Tweet'] = df['Tweet'].str.replace("\xe2\x80\x99\xe2\x80\xa6", " ", False)

    df = df.drop(['name', 'username', 'tweetid', 'time', 'tweets'], axis=1)

    df.to_csv('DF1.csv')
```

Figure 5.3: Data Cleaning Function

In addition to removing the noise other the following procedures were carried out;

i. Specification of all characters within the tweet to be of type string.
ii. Removal of emoticons
iii. Removal of columns that are of no interest to the research.

5.1.3 Data Labelling and Merging

Cleaned data from the two initial datasets were imported into a Pandas data frame and labeled then merged into one dataset for later use in building the various classifiers of interest. The ISIS Fanboy Tweets were labeled as **0** to represent instances of radical tweet posts; tweets previously labeled as **4** in the Sentiment 140 dataset were labeled as **1** to represent instances of non-radical tweet posts. To ensure a relative distribution of the two classes all 17,000 instances from the ISIS Fanboy data were used and an estimated equal number of instances from the Sentiment 140 data. The merged dataset was then randomized and checked for duplicate instances. The final dataset had around 34,000 instances labeled as either 1 or 0.

5.2 Model Training

After the pre-processing stage, the outcome is two main objects i.e. a csv file with labeled data and a word embedding file saved as a text file. These two are both used in the development of machine learning models. The labeled csv file is imported into a Pandas data frame which is a two-dimension data structure consisting of rows and columns. The data is then latter split i.e. the first split of 80% was used to train the model and the remainder 20% was used for testing the model.

5.2.1 Model Architecture

The architecture of the proposed model had four main parts i.e.

i. **Word Embedding Layer**

This a representation of text where words that have the same meaning have a similar representation.

ii. **Deep network Layer**

This takes the sequence of embedding vectors as input and feeds them into the RNN.

iii. **Fully Connected Layer**

This takes the deep representation from the RNN/LSTM/GRU and transforms it into the final output classes or class scores. This component is comprised of fully connected layers along with batch normalization and optionally dropout layers for regularization.

iv. **Output Layer**

With regards to this research this a Sigmoid activation was used since there were only two possible classes.

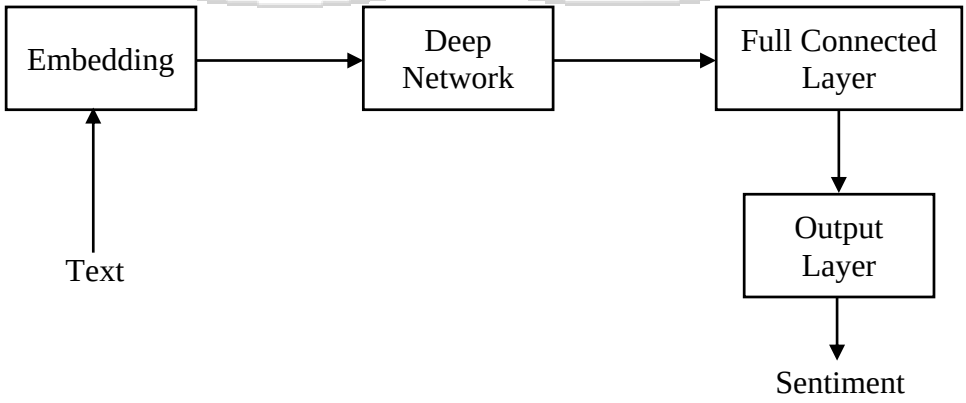


Figure 5.4: Proposed RNN(LSTM) Model Architecture

5.2.2 Creating Word Embeddings

This research opted to first separately learn word embeddings for each word within the corpus and then pass these to the embedding layer within the neural network. This technique

would also allow the use of other/any pre-trained word embedding hence can be a point of carrying out experiments i.e. experimenting with different vocabularies to check how the model performance is affected by each. To achieve this, the *Gensim* python library which is an implementation of Word2Vec (Ali, 2019). This was done in two major steps i.e.

i. Preprocessing all collected data.
ii. Passing the tokens to the word2vec algorithm to learn an embedding for each word

Figure 5.5 below shows a code snippet and description of various parameters used when learning the word embeddings.

```
import gensim

vocabulary = gensim.models.Word2Vec(sentences=tweet_lines, size=200, window=5, workers=4, min_count=1)
words = list(vocabulary.wv.vocab)
print('Vocabulary size:  %d'  % len(words))
```

Figure 5.5 Learning Embeddings For-Loop

Sentences: List of sentences i.e. the tweets containing words from which the embedding needs to be learned. Size: The number of dimensions for which each word will be represented i.e. 200 for this research.

As a precaution, the embeddings were checked to see how, well it could identify similar or related words. The figure below shows the testing of words similar to *'attack'*. The first result is *'coalition'* which implies the words *'attack'* and *'coalition'* are closely related and likely to appear closely in a sentence.

```
vocabulary.wv.most_similar('attack')
```

```
[('coalition', 0.9970760345458984),
 ('near', 0.9970357418060303),
 ('shaer', 0.9970274567604065),
 ('isis', 0.9969481229782104),
 ('positions', 0.9967876672744751),
 ('iraq', 0.9966533780097961),
 ('forces', 0.996362566947937),
 ('damascus', 0.9963335394859314),
 ('air', 0.9959625005722046),
 ('battles', 0.995651364326477)]
```

Figure 5.6: Testing Word Similarities

5.2.3 Training the Model

This work made use of an implementation of the Sequential model provided by Python's Keras machine learning library. Figure 5.7 shows the python implantation and summary details of the model during development.

```
from keras.models import Sequential
from keras.layers import Dense, Embedding, LSTM
from keras.layers.embeddings import Embedding
from keras.initializers import Constant


model = Sequential()
embedding_layer = Embedding(num_words,
                            embedding_dimension,
                            embeddings_initializer = Constant(embedings_matrix),
                            input_length=max_lenght,
                            trainable = False)

model.add(embedding_layer)
model.add(LSTM(32, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Figure 5.7: Model Summary

5.3 Model Validation

As mentioned earlier, 20% of the data was set aside for testing. The parameters used to assess the model's performance were: Accuracy, Loss, Precision, and F-Score. A confusion matrix was produced to describe various values that were used to calculate the various performance parameters.

Table 5.1: Confusion Matrix

	Actual 0	Actual 1
Predicted 0	3258	209
Predicted 1	172	3259

Values of true positive, false positive, true negative and false negative were obtained from Table 5.1 and depicted in Table 5.2.

Table 5.2: Classification Values

True Positive (Non-Radical)	3259
False Positive (Non-Radical)	209
True Negative (Radical)	3258
False Negative (Radical)	172

Figure 5.8 below illustrates a graphical representation of the confusion matrix done using Python Matplotlib library.

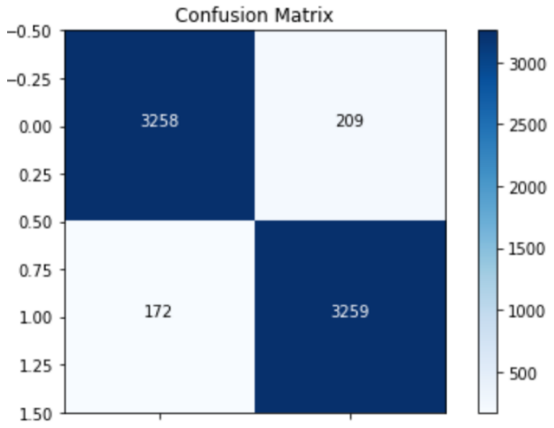


Figure 5.8: Confusion Matrix Plot

The performance metrics were calculated as per the formulas described in Chapter 3 and depicted in Table 5.3 below:

Table 5.3 Performance of the RNN (LSTM) Model

Accuracy	F-Score	Precision
0.95	0.94	0.94

The following graphical visualization i.e. Figure 5.9 was produced to illustrate the variation of the Accuracy of both training and testing data during the eight different iterations (epochs) of the training process.

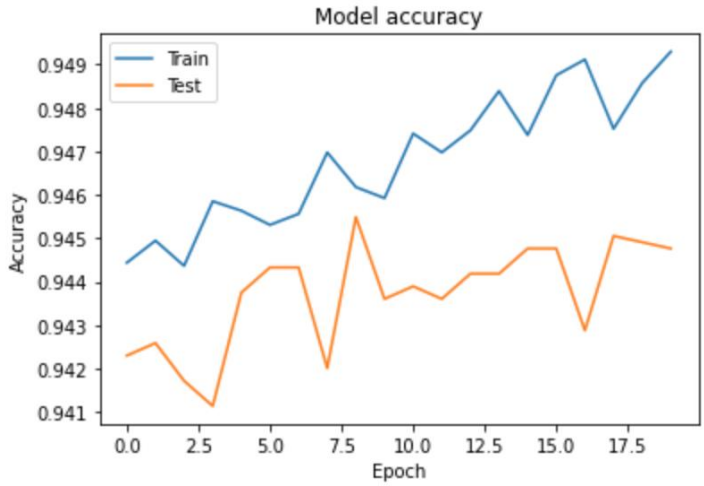


Figure 5.9: Model Accuracy Visualization

The following graphical visualization i.e. i.e. Figure 5.10 was produced to illustrate the variation of the Loss both training and testing data during the eight different iterations (epochs) of the training process.

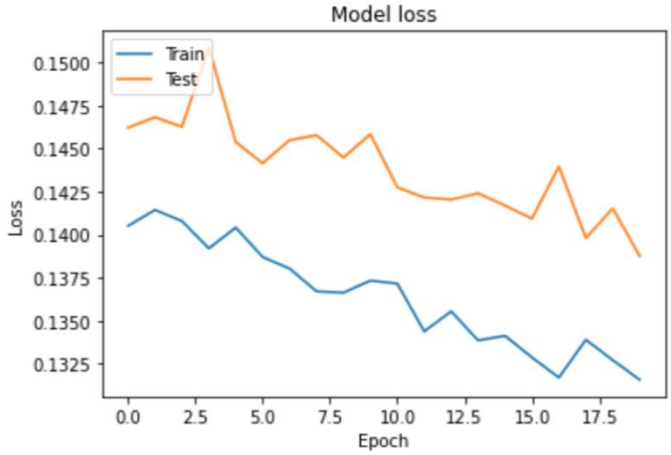


Figure 5.10: Model Loss Visualization

5.4 Using the Model in Prediction

To use the model for prediction test cases were passed as strings then each was transformed into corresponding vector representations before being fed into the model. To avoid having to train the model each time a prediction instance needs to be made, it was exported into a file with an *h5* extension which is suitable for storing the structure of the neural network. This was done using the *load_model* function from the Keras library and is depicted in Figure 5.11 below.

```
from keras.models import load_model

model.save('drive/My Drive/MS Project/Main/Models/main_lstm_model.h5')
lstm_model = load_model('drive/My Drive/MS Project/Main/Models/main_lstm_model.h5')
```

Figure 5.11: Saving and Loading LSTM Model

The test cases were first tokenized, then padded to ensure their vector representations were the same as those used when training the model. Figure 5.12 shows the sample test cases and how they were tokenized and padded before being fed into the model for prediction.

```
[50] test1 = 'I love the Jihad ideology maybe we need to review it and give Jihad fighters
     test2 = 'Still no radical statement about the new virus.'
     test3 = 'Saying the word radical dosent make you radical'
     test4 = 'Life is too short to be boring, have a good day and support the Jihad'
     test5 = ''

[51] test_samples = [test1, test2, test3, test4, test5]
     test_samples_tokens = tokenizer_object.texts_to_sequences(test_samples)
     test_samples_paddings = pad_sequences(test_samples_tokens,  maxlen=max_lenght)

     lstm_model.predict(x=test_samples_paddings)
```

Figure 5.12: Sample String Test Cases

The result of each prediction was a float value between 0 and 1. These values were further rounded off to either be integers valued 0 or 1; any value equal to or greater than 0.5 was considered Non -Radical and any below 0.5 was rounded to 0 i.e. Radical. Figure 5.13 shows both the initial float prediction values and rounded-off values.

```
[52] lstm_model.predict(x=test_samples_paddings)

array([[0.06538881],
       [0.57258385],
       [0.7723924 ],
       [0.03132489],
       [0.3017333 ]], dtype=float32)

test_rounded_predictions = lstm_model.predict_classes(test_samples_paddings)
test_rounded_predictions

array([[0],
       [1],
       [1],
       [0],
       [0]], dtype=int32)
```

Figure 5.13: Test Case Float and Integer Results

Once texts have been assigned a class, other metadata such as the location of the post, time of post, and user handles can be used to narrow down on online posts thus filter out personal posting radical material geared towards radicalizing other online users. To obtain tweets and use the model for prediction of tweets as a third party, the below steps apply:

i. Create a Twitter Developer Account
ii. Create a Streaming App to get Use Credentials
iii. Use guides to stream and classify desired Tweets i.e. One specifies A User Account Handle or Geo-Location to obtain Tweets to classify
iv. Pass the tweets to a cleaning function and then export as a csv.
v. Read-in the csv into a data frame, perform tokenization and padding then feed the padded vector sequences into the model for prediction.

Chapter 6 Experiments and Discussions

6.1 Introduction

The first section of this chapter goes over various experiments to allow for comparison of the performance of the developed LSTM model and other machine learning models used in the field of natural language processing specifically sentiment analysis of Twitter data. In addition to the performance values, instances of labeled data were randomly selected from the corpus and fed into each of the classifiers to test if the predicted result were correct. The second section covers discussions of the experiment results as well as a conclusion on the best set of parameters to use when developing a similar model.

6.2 Experiments

6.2.1 Experiment 1: Using Different Classifiers

The first validation experiment was aimed at comparing the performance of the developed LSTM model with other machine learning approaches i.e. Support Vector Machine and Naïve Bayes. Both approaches use BOW feature generation as opposed to the Embedding feature representation which this work relied on for generating features. The same dataset used in developing the LSTM model was used with a similar split for training and testing.

SVM

As described in chapter two, SVM is a supervised machine learning algorithm applicable for solving classification and/or regression problems. A linear SVM classifier was developed using tf-idf weighting with a bigram feature type. The performance results were quite good, however, the results become inconsistent with every new run especially when the data was shuffled before training the model in each new run.

Naïve Bayes

This approach shared similarities with the SVM method since similar libraries, vectorization techniques, and feature weighting were used in the development of the classifier. The Naïve Bayes model performed significantly well and did not experience an inconsistency when the data was shuffled in each new iteration of training and testing.

6.2.2 Experiment 2: Using Different Word Vocabularies

The second validation experiment aimed at testing how different vocabularies trained/developed by third-party datasets can be used in the prediction model. In this case, a model highly similar to that developed in Chapter five was created, however, the embedding file developed in experiment one was used instead of the one used in chapter five. This meant the number of words represented reduced from 44899 to 1190. Figure 6.3 shows the Accuracy and Loss values across different epochs during model development.

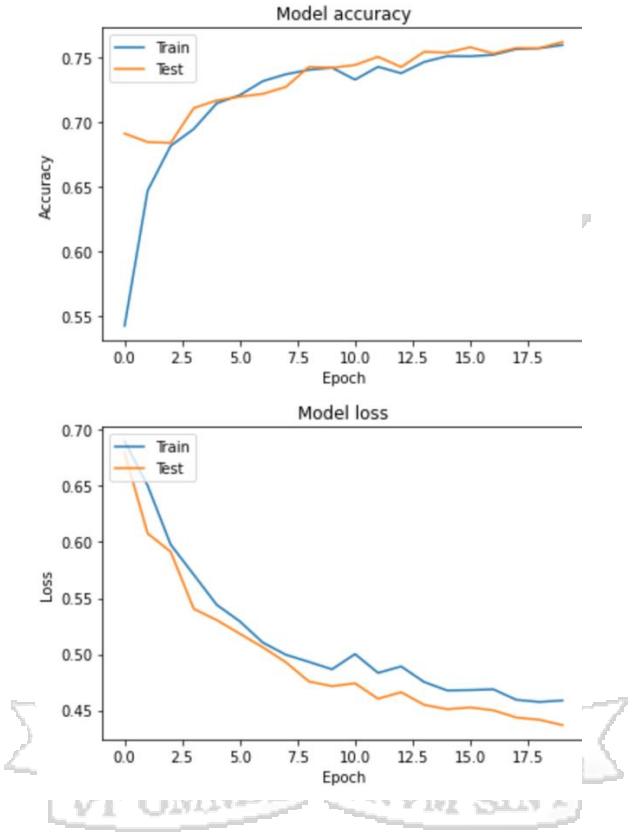


Figure 6.1: Experiment Two Performance

6.2.3 Experiment 3: Using Kenyan Data

In this experiment a model was developed using a similar parameter describes in chapter five but with a different dataset used for training and testing. This was aimed at showing the concept of this can be implemented as long as the corpus used is created using similar techniques. The LSTM layer then controls the flow of information

6.2.1.1 Corpus Creation

To build an adequate corpus for this experiment, the research streamed tweets using keywords and geotags as filtration parameters to ensure the data collected was from the location of interest i.e. Kenya. Additionally, self-generated statements were appended to the document by the researcher, this was done to try and achieve an equal class distribution for the dataset to be used in model development. This data was then preprocessed using similar steps described in Chapter five. The data was then stored as a csv file containing various instances of text. Labeling was done manually by individually assigning radical and non-radical labels to each based upon the researcher's knowledge. Figure 6.1 shows a snippet of the data with both the text and label attributes.

	Label
wmnjoya moh kenya kenyagovernors knh ho...	1
moh kenya corona virus temperature ...	1
gabrieloguda sicily kanini kariuki the m...	1
route ntsa kenya npsofficial how...	1
niliambia mum mimi nitaenda niungane wale wah...	0

Figure 6.2: Experiment Three Corpus

6.2.1.2 Training and Testing

The labeled csv file was imported into a Pandas data frame split i.e. 80% for training and 20 % for testing. A vocabulary was also trained using all the data available for this experiment and the resulting word count was 1190 which was notably less than that used in the main model in Chapter five.

6.2.1.3 Results

After 20 passes through over the corpus during training the following graph was produced to visualize the model Accuracy and Loss.

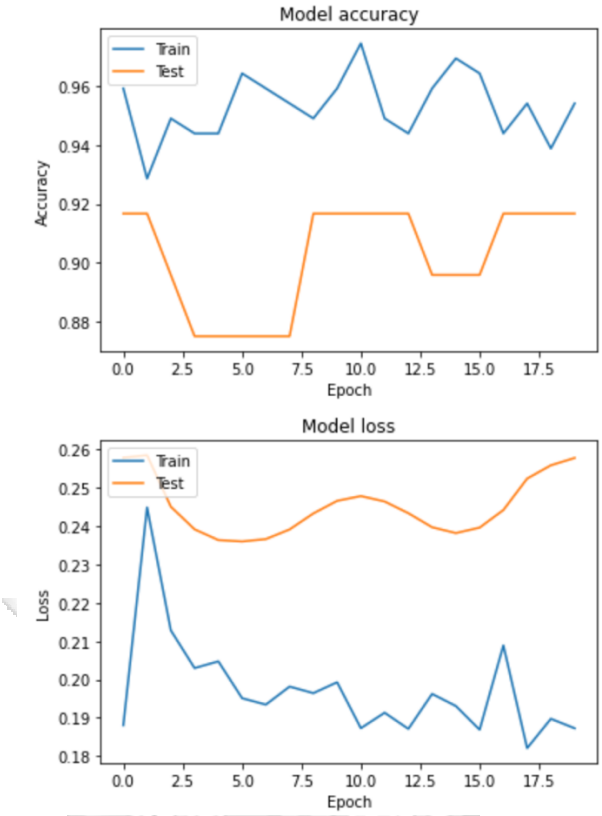


Figure 6.3: Experiment Three Performance

6.2.4 Classification Comparisons

After setting up all the experiment models, a select number of instances with known classes were selected from the dataset and tested on all four models to compare how each classified an instance regardless of each individual's model's accuracy values. Table 6.1 summarizes the results of this experiment. The values of prediction were rounded off to be either 1 (Non-Radical) or 0 (Radical).

Table 6.1: Test Classification Results

Sample Statement	Actual Class	LSTM Predicted Class	SVM Predicted Class	NB Predicted Class
I love the Jihad ideology maybe we need to review it and give Jihad fig hters a chance to share their side of the conversation	0	0	0	0
Still no radical statement about the new virus	1	1	1	0
Saying the word radical doesn't make you radical	1	1	1	0
Life is too short to be boring, have a good day and support the Jihad	0	0	1	1

6.3 Discussions

6.3.1 Developed Model

Once a tweet has been obtained for classification, the first step taken is noise removal to only retain relevant parts of the tweet needed for classification. A cleaned text is the padded and tokenized to generate padded sequence values that are fed as weights into the neural network. By referencing the pre-trained vocabulary, the model then infers relationships i.e. context of words within the padded sequences; this is done in the network's embedding layer. The LSTM and Dense layers then control information flow to classify instance; Sigomid activation is used as this exists in the range 0 to 1.

6.3.2 Validation Experiment Results

The first experiments tested other machine learning approaches/algorithms used in the field of sentiment analysis. The SVM and Naïve Bayes approaches averaged an accuracy score of 95%. However, when tested with the random instance from the corpus, the SVM model wrongly classified a statement in some instances while the Naïve Bayes model maintained an accurate classification in most cases. The wrong classification of instances by the SVM and Naïve Bayes models could be a result of the model considering word count in classification. The SVM model relies on word count when determining the class of a given text thus the context of each word in the text is not taken into consideration; as a result, this occasionally leads to inaccurate classifications. This goes to show that considering context during classification can help improve the classification result.

Chapter 7 Research Conclusions and Recommendations

7.1 Conclusion

This research intended to develop an automated tool that could be able to detect radicalization on Twitter using machine learning techniques. To enable successful execution of the research it was necessary to understand the nature of radicalization by the various radical group on Twitter. To obtain relevant literature on existing techniques used, various publications and articles were reviewed to provide needed knowledge for this particular research. Additionally, machine learning methods in the field of text classification specifically sentiment analysis were reviewed to understand the different stages in the processes as well as potential areas for improvement. Based on this review an embedding feature representation technique was chosen for feature generation. To improve on the high dimensional feature vectors exhibited by the BOW alternative feature generation technique when dealing with large size texts, padding was introduced to limit token vectors sequences to a static value and have them in the same shape and size before using them as weights as this a requirement for artificial neural networks. After data was collected and pre-processed, a 200-dimensional vocabulary was created to aid in referencing context between words based on their vector representation. This vocabulary was later used in the first embedding layer of the classification model. An LSTM model was then developed to perform text(tweet) classification and return a score value in range 0 to 1 that can be used to infer a class. The accuracy of the developed model was 95% which given the even distribution of samples in the two classes indicates a high ratio for correctly classified texts in the test data to the total number of texts.

As a means of justifying this research's approach, two experiments were carried out to compare other known approaches in the field of sentiment analysis i.e. use of Support Vector Machine and Naïve Bayes models. These experiments demonstrated how the proposed approach using neural networks in conjunction with word embedding feature representations as well as a pre-trained vocabulary to infer context produced much more consistent results. The experiments also showed potential areas of customization when developing the model during the feature representation stage where third party embeddings can be used without severely degrading the performance of the classifiers.

7.2 Recommendations

This research showed that the use of artificial neural networks in the field of sentiment analysis can provide a means to automatically perform text classification that is useful in detecting radicalization on Twitter. This provides a faster means as compared to the existing radicalization detection techniques which rely heavily on human supervision or rely on the use of the bag of words approach for feature representation. However, the developed model still has the potential to improve in terms of classification accuracy. This could be done by creating a vocabulary of almost all possible words in various languages and a corresponding embedding file to represent how the words in this vocabulary would relate to each other. As per the Oxford dictionary, there are only 171,476 unique words; this research only used a vocabulary of around 45,000 words which goes to show that increase in vocabulary size would enhance performance by allowing the model to learn additional context information for existing as well as new words within the expanded vocabulary. Additionally, context can be inferred using emoticons also commonly referred to as *'emojis'*. The use of emoticons is generally aimed to increase tweet post/account engagements (Window, 2019). This new form of data in tweets can thus be used to improve context detection.

7.3 Future Works

Radical statements can be expressed in a variety of languages and are highly dependent on the location where a radical grouped is based; this study limited it's language scope to English. Future research can take advantage of open source tools namely; Open Machine Translation to aid in growing corpus size. Additionally, exploring the use of multiple languages when building a word vocabulary can also aid in increasing the number of words represented within the embedding layer thus provide a basis to carry out sentiment analysis of statements constructed using more than one language e.g. *Sheng* which is a type of language popular in Kenya that allows one to combine Swahili and English when constructing sentences. Finally, identification and incorporation of slang i.e. informal commonly used language can also help in enhancing classifier flexibility to work with multiple languages.

References

AFP. (2014, June). *How Isis used Twitter and the World Cup to spread its terror*. Retrieved May 2019, from The Telegraph: http://www.telegraph.co.uk/news/worldnews/middleeast/iraq/10923046

Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. (2011). Sentiment Analysis of Twitter Data. 30-38.

Ali, Z. (2019, January 6). *Word2vec Tutorial*. Retrieved January 2020, from Medium: https://medium.com/@zafaralibagh6/a-simple-word2vec-tutorial-61e64e38a6a1

Almagor, R. C. (2012). In the internet's way: Radical, terrorist Islamists on the free highway. *International Journal of Cyber Warfare and Terrorism, 2*(3).

Archetti, C. (2015). Terrorism, Communication and New Media: Explaining Radicalization in the Digital Age.

Bartlett, J., & Fisher, A. (2015, April 1). *How To Beat The Media Mujahideen*. Retrieved June 2019, from Vox Pol: https://www.voxpol.eu/how-to-beat-the-media-mujahideen/

Bartosz, G. (2018, March 6). *The TF*IDF Algorithm Explained*. Retrieved June 2019, from Onely: https://www.onely.com/blog/what-is-tf-idf/

Bilgen, A. (2012, July 22). *Terrorism and the Media: A Dangerous Symbiosis*. Retrieved April 2019, from https://www.e-ir.info/author/arda-bilgen/

Boot, M. (2015, March 16). *Why social media and terrorism make a perfect fit*. Retrieved May 2019, from The Washington Post: https://www.washingtonpost.com/opinions/2019/03/16/why-social-media-terrorism-make-perfect-fit/?noredirect=on&utm_term=.b9f26dfd8651

Brownlee, J. (2019, August 12). *A Tour of Machine Learning Algorithms*. Retrieved September 2019, from Machine Learning Mastery: https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

Cam-Stei, D. (2019, February 12). *Word Embedding Explained, a comparison and code tutorial*. Retrieved 2019, from Medium: https://medium.com/@dcameronsteinke/tf-idf-vs-word-embedding-a-comparison-and-code-tutorial-5ba341379ab0

Cat, S. (2017, April 29). *Boosting algorithm: AdaBoost*. Retrieved April 2019, from Towards Data Science: https://towardsdatascience.com/boosting-algorithm-adaboost-b6737a9ee60c

Chawla, J. S. (2017, October). *Word Vector Representations*. Retrieved January 2020, from Medium: https://medium.com/@japneet121/introduction-713b3d976323

Chen, F. (2019, November 27). Knowledge-enhanced neural networks for sentiment analysis of Chinese reviews. *Neurocomputing, 368*, 51-58.

Chia, D. (2018, December 6). *An implementation guide to Word2Vec*. Retrieved January 2020, from Towards Data Science: https://towardsdatascience.com/an-implementation-guide-to-word2vec-using-numpy-and-google-sheets-13445eebd281

Creswell, J. W. (2014). *Research Design.* California: Sage Publishers.

Delphine, M., & Camille, S. (2020). *Understanding radicalisation*. Retrieved February 2020, from EIP.

Eibe, F., & Witten, I. H. (2005). *Data Mining: Practical machine learning tools and techniques.* Morgan Kaufmann.

Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., & Lehmann, S. (2017). Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *Conference on Empirical Methods in Natural Language Processing.*

Feldman, R., & Sanger, J. (2006). *Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data.* New York: Cambridge University Press.

Fisher, A., & Prucha, N. (2014). The Call-Up. *Combating Terrorism Exchange, 4*(3), 73. Retrieved June 2019, from The Call-Up: https://globalecco.org/the-call-up-the-roots-of-a-resilient-and-persistent-jihadist-presence-on-twitter

Gajare, S. (2019, August 25). *Why Feature Selection?* Retrieved January 2020, from https://medium.com/analytics-vidhya/why-feature-selection-144816f05ee8

Gandhi, R. (2018, May 5). *Naive Bayes Classifier*. Retrieved April 2019, from Towards Data Science: https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c

Ganesan, K. (2019). *Text Pre-processing for NLP*. Retrieved from KD Nuggets: https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html

Geitgey, A. (2018, August). *Text Classification is Your New Secret Weapon*. Retrieved January 2020, from Medium: https://medium.com/@ageitgey/text-classification-is-your-new-secret-weapon-7ca4fad15788

Gensim. (n.d.). Retrieved January 2020, from Python Package Index: https://pypi.org/project/gensim/

Ghiassi, M., Arnuado, P., & Moon, B. (n.d.). Automated text classification using a dynamic artificial neural network model. *Expert Systems with Applications, 39*(12), 10967-10976.

Go, A., Bhayani, R., & Huang, L. (2009). Twitter Sentiment Classification using Distant Supervision.

Gupta, S. (2018, January 7). *Sentiment Analysis: Concept, Analysis and Applications.* Retrieved March 2019, from https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17

Hill, T., & Lewicki, P. (2007). *Statistics: Methods and Applications* (Vol. 1). StatSoft. Retrieved March 2019, from http://www.statsoft.com/Textbook

How ISIS Uses Twitter. (2015). (Kaggle) Retrieved December 2019, from Kaggle: https://www.kaggle.com/fifthtribe/how-isis-uses-twitter

How ISIS Uses Twitter. (2016, May 13). Retrieved November 2019, from Kaggle: https://www.kaggle.com/fifthtribe/how-isis-uses-twitter

Huang, C., Fu, T., & Chen, H. (2010, April). Text-based video content classification for online video-sharing sites. *Journal of the Association for Information Science and Technology, 61*(5), 891-906.

Jefferson, H. (2018, November 21). *Get Old Tweets Programatically*. Retrieved June 2019, from https://github.com/Jefferson-Henrique/GetOldTweets-python

Joachims, T. (1998). Text categorization with Support Vector Machines: Learning with many relevant features.

Klausen, J. (2015). Tweeting the Jihad: Social Media Networks of Western Foreign Fighters in Syria and Iraq, Studies in Conflict & Terrorism. *Studies in Conflict & Terrorism, 38*(1), 1-22.

Kosoff, M. (2017, September 20). *Is Twitter Winning It's War On Terrorism?* Retrieved March 2019, from https://www.vanityfair.com/news/2017/09/is-twitter-winning-its-war-on-terrorism

Kulshrestha, R. (2019, November 24). *Word2Vec: Skip-gram and CBOW*. Retrieved January 2020, from Medium: https://towardsdatascience.com/nlp-101-word2vec-skip-gram-and-cbow-93512ee24314

Lee, K., Palestia, D., Narayanan, R., Agrawal, A., & Choudhary, A. (2011). Twitter Trending Topic Classification .

Liu, B. (2007). *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data.* London: Springer Verlag .

Magogo, S. (2017, November). The Effectiveness of Counterterorsim Strategies in Kenya.

Mariette, A., & Khanna, R. (2015). Theories, Concepts, and Applications for Engineers and System Designers. In *Efficient Learning Machines* .

Mittal, A. (2019, October 12). *Understanding RNN and LSTM*. Retrieved January 2020, from Towards Data Science: https://towardsdatascience.com/understanding-rnn-and-lstm-f7cdf6dfc14e

Naz, R., & Khan, M. (2015). Rapid applications development techniques: A critical review. *IJSEIA*(9), 163-176.

Ngoge, L. A. (2016). *Real – time sentiment analysis for detection of terrorist activities in Kenya*. Retrieved November 2019, from http://su-plus.strathmore.edu/handle/11071/4826

Olah, C. (2015, August 27). *Understanding LSTM Networks*. Retrieved January 2020, from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Pandey, P. (2019, November 25). *Data Preprocessing Concepts*. Retrieved January 2020, from Towards Data Science: https://towardsdatascience.com/data-preprocessing-concepts-fa946d11c825

Patel, S. (2017, May 3). *SVM Theory*. Retrieved March 2019, from Medium: https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72

Pedregosa, F., Michel, V., Thirion, B., Gramfort, A., & Varoquaux, G. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 2825-2830.

Peng, F. (2003). *Augmenting Naive Bayes Classifiers with Statistical Language Models*. Retrieved July 2019, from Computer Science Department Faculty Publication Series: https://scholarworks.umass.edu/cs_faculty_pubs/91/?utm_source=scholarworks.umass.edu%2Fcs_faculty_pubs%2F91&utm_medium=PDF&utm_campaign=PDFCoverPages

Pyle, D. (1999). Data Preparation for Data Mining.

Rathor, S. (2018, June 2). *Simple RNN vs GRU vs LSTM*. Retrieved January 2020, from Medium: https://medium.com/@saurabh.rathor092/simple-rnn-vs-gru-vs-lstm-difference-lies-in-more-flexible-control-5f33e07b1e57

Reddy, V. (2018, November 12). *Preparing the text Data with scikit-learn*. Retrieved July 2019, from https://medium.com/@vasista/preparing-the-text-data-with-scikit-learn-b31a3df567e

Reddy, V. (2018, November). *Sentiment Analysis using SVM*. Retrieved July 2019, from Medium: https://medium.com/@vasista/sentiment-analysis-using-svm-338d418e3ff1

Saul, H. (2015, March 6). *ISIS Propaganda*. Retrieved April 2019, from Independent: https://www.independent.co.uk/life-style/gadgets-and-tech/isis-propaganda-study-finds-up-to-90000-twitter-accounts-supporting-extremist-group-10090309.html

Security Council Adopts Resolution . (2014, August 15). Retrieved May 2019, from United Nations: https://www.un.org/press/en/2014/sc11520.doc.htm

Selamat, A., & Zainuddin, N. (2014). Sentiment Analysis Using Support Vector Machine. *International Conference on Computer, Communication, and Control Technology.* Kedah.

Sentiment 140. (n.d.). Retrieved January 2020, from http://help.sentiment140.com/for-students/

Shane, S., & Hubbard, B. (2014). *ISIS Displaying a Deft Command of Varied Media*. Retrieved June 2019, from https://www.nytimes.com/2014/08/31/world/middleeast/isis-displaying-a-deft-command-of-varied-media.html

Singh, N., Devi, M., & Anjana, M. K. (2017). Document representation techniques and their effect on the document Clustering and Classification: A Review. *International Journal of Advanced Research in Computer Science, 8*(5), 1780-1784.

Singh, P. (2019, September 4). *Fundamentals of Bag Of Words and TF-IDF*. Retrieved January 2020, from Medium: https://medium.com/analytics-vidhya/fundamentals-of-bag-of-words-and-tf-idf-9846d301ff22

Smetanin, S. (2018, September 1). *Sentiment Analysis of Tweets using Multinomial Naive Bayes*. Retrieved July 2019, from https://towardsdatascience.com/sentiment-analysis-of-tweets-using-multinomial-naive-bayes-1009ed24276b

Spacy. (2019, September). Retrieved September 2019, from Github: https://github.com/explosion/spaCy

Springer, D. R. (2009). Islamic Radicalism and Global Jihad.

Stehman, S. V. (1997, October). Selecting and interpreting measures of thematic classification accuracy. *Remote Sensing of Environment, 69*(1), 77-89.

Tch, A. (2017, August 4). *The mostly complete chart of Neural Networks, explained*. Retrieved January 2020, from Towards Data Science: https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464

Turner, L. (2018). The Path to Terrorism: The Islamic State and Its Recruitment Strategies.

Twitter. (2019, 7 10). *Twitter About*. Retrieved from Twitter: https://about.twitter.com/

UNODC Education for Justice. (2018, July). *Radicalization' and Violent Extremism*. Retrieved February 2020, from https://www.unodc.org/e4j/en/terrorism/module-2/key-issues/radicalization-violent-extremism.html

Upender, M. (2018, October 14). *Confusion Matrix Terminology*. Retrieved from Medium: https://medium.com/@upenderreddymuddasani/confusion-matrix-terminology-aeb443e01079

Weber, I., Darwish, K., & Magdy, W. (2016, February). *Using Twitter to study the antecedents of ISIS support*. Retrieved April 2019, from https://firstmonday.org/ojs/index.php/fm/article/view/6372/5194#author

Weng, J. (2019, August). Retrieved January 2020, from Towards Data Science: https://towardsdatascience.com/nlp-text-preprocessing-a-practical-guide-and-template-d80874676e79

Window, M. (2019, January). *Ways to increase your Tweet engagements* . Retrieved April 2020, from Twitter Business: https://business.twitter.com/en/blog/5-ways-to-increase-your-tweet-engagements-with-emojis.html

Yining, Y., & Pedersen, J. O. (1997). A Comparative Study on Feature Selection in Text Categorization. *International Conference on Machine Learning 1997*, 412-420.

Zarella, D. (2009). *The Social Media Marketing Book.* O Reilly Media.

APPENDIX A: Originality Report

Sentiment Analysis Model for Detection of Radicalization on Twitter: Embedding Feature Representation based Approach

By

Emmanuel Oluoch Olang'

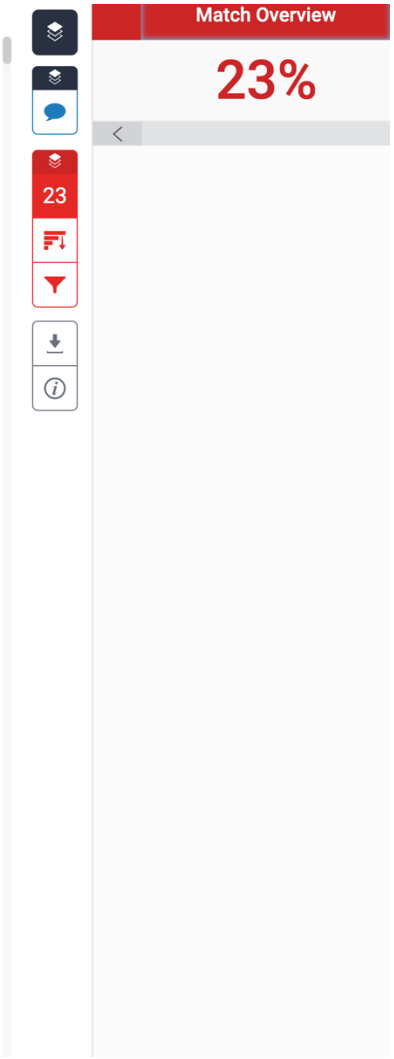
114899

A Research Thesis Submitted to the Faculty of Information in partial fulfillment of the requirements for the award of the degree of Master of Science Information Technology (MSc. IT) at Strathmore University.

Master of Science Information Technology

Strathmore University

April 2020



APPENDIX B: Python Programs

Main LSTM Model (Training)

```
import pandas as pd
import numpy as np

df = pd.DataFrame()
df = pd.read_csv('drive/My Drive/MS Project/Main/Data/data.csv')
df.Tweet = df.Tweet.astype(str)
df.head(3)

"""Train/Generate Vocabulary (Word Vectors)"""

import csv
from nltk.stem.porter import *
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import string
import nltk
nltk.download('stopwords')
nltk.download('punkt')

tweet_lines = list()
lines = df['Tweet'].values.tolist()

for line in lines:
    tokens = word_tokenize(line)

    tokens = [w.lower() for w in tokens]

    table = str.maketrans('', '', string.punctuation)
    stripped = [w.translate(table) for w in tokens]

    words = [word for word in stripped if word.isalpha()]

    stop_words = set(stopwords.words('english'))
    words = [w for w in words if not w in stop_words]
    tweet_lines.append(words)

print(len(tweet_lines))

import gensim

vocabulary = gensim.models.Word2Vec(sentences=tweet_lines, size=200, window=5, workers=4, min_count=1)
words = list(vocabulary.wv.vocab)
print('Vocabulary size:  %d'  % len(words))

vocabulary.wv.most_similar('attack')

# export the vocabulary word embedings into a txt file for use in model traning later

filename = 'drive/My Drive/MS Project/Main/vocabulary_embeddings.txt'
vocabulary.wv.save_word2vec_format(filename, binary=False)

"""Train the actual lstm model to classify radical statements"""

df = df.drop(df.index[1])
X_train = df.loc[:17245, 'Tweet'].values
y_train = df.loc[:17245, 'Label'].values
X_test = df.loc[17246:, 'Tweet'].values
y_test = df.loc[17246:, 'Label'].values
```

```
from tensorflow.python.keras.preprocessing.text import Tokenizer
from tensorflow.python.keras.preprocessing.sequence import pad_sequences

tokenizer_object = Tokenizer()
total_tweets = X_train + X_test
tokenizer_object.fit_on_texts(total_tweets)

#padding
max_lenght = max([len(s.split()) for s in total_tweets])

#vocab size
vocab_size = len(tokenizer_object.word_index) + 1


X_train_tokens = tokenizer_object.texts_to_sequences(X_train)
X_test_tokens = tokenizer_object.texts_to_sequences(X_test)

X_train_pad = pad_sequences(X_train_tokens, maxlen=max_lenght, padding='post')
X_test_pad = pad_sequences(X_test_tokens, maxlen=max_lenght, padding='post')
```

```
import os

embedings_index = {}
f = open(os.path.join('', 'drive/My Drive/MS Project/Main/vocabulary_embeddings.txt'), encoding="utf-8")
for line in f:
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:])
    embedings_index[word] = coefs
f.close()

tokenizer_obj = Tokenizer()
tokenizer_obj.fit_on_texts(tweet_lines)
sequences = tokenizer_obj.texts_to_sequences(tweet_lines)

word_index = tokenizer_obj.word_index
print(len(word_index))

tweet_pad = pad_sequences(sequences, maxlen=max_lenght)
label = df['Label'].values
print(tweet_pad.shape)
print(label.shape)

embedding_dimension = 200
num_words = len(word_index) + 1
embedings_matrix = np.zeros((num_words, embedding_dimension))

for word, i in word_index.items():
    if i > num_words:
        continue
    embedings_vector = embedings_index.get(word)
    if embedings_vector is not None:
        embedings_matrix[i] = embedings_vector

print(num_words)

from keras.models import Sequential
from keras.layers import Dense, Embedding, LSTM
from keras.layers.embeddings import Embedding
from keras.initializers import Constant
```

```
model = Sequential()
embedding_layer = Embedding(num_words,
                            embedding_dimension,
                            embeddings_initializer = Constant(embedings_matrix),
                            input_length=max_lenght,
                            trainable = False)

model.add(embedding_layer)
model.add(LSTM(32, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

model.summary()

validation_split = 0.2

indices = np.arange(tweet_pad.shape[0])
np.random.shuffle(indices)
tweet_pad = tweet_pad[indices]
label = label[indices]


num_validation_samples = int(validation_split * tweet_pad.shape[0])

X_train_pad = tweet_pad[:-num_validation_samples]
y_train = label[:-num_validation_samples]
X_test_pad = tweet_pad[-num_validation_samples:]
y_test = label[-num_validation_samples:]

history = model.fit(X_train_pad, y_train, batch_size=128, epochs=20, validation_data=(X_test_pad, y_test), verbose=2)

import matplotlib.pyplot as plt

plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

# Plot training & validation loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

from keras.models import load_model

model.save('drive/My Drive/MS Project/Main/Models/main_lstm_model.h5')
lstm_model = load_model('drive/My Drive/MS Project/Main/Models/main_lstm_model.h5')
```

Prediction Module

```
sample_text1 = ''
sample_text2 = ''
samples = [sample_text1,sample_text2]
samples_tokens = tokenizer_object.texts_to_sequences(samples)
samples_paddings = pad_sequences(samples_tokens,  maxlen=max_lenght)
samples_rounded_predictions = lstm_model.predict_classes(samples_paddings)
print(samples_rounded_predictions)
```

APPENDIX C: Ethical Approval

Strathmore UNIVERSITY

28th November 2019

Mr Olang', Emmanuel
emmanuel.olang@strathmore.edu

Dear Mr Olang',

RE: Sentiment Analysis for Detection of Radicalization on Twitter

This is to inform you that SU-IERC has reviewed and **approved** your above research proposal. Your application approval number is **SU-IERC0570/19** .The approval period is **28th November, 2019 to 27th November, 2020.**

This approval is subject to compliance with the following requirements:

i. Only approved documents including (informed consents, study instruments, MTA) will be used
ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by SU-IERC.
iii. Death and life threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to SU-IERC within 72 hours of notification
iv. Any changes, anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to SU-IERC within 72 hours
v. Clearance for export of biological specimens must be obtained from relevant institutions.
vi. Submission of a request for renewal of approval at least 60 days prior to expiry of the approval period. Attach a comprehensive progress report to support the renewal.
vii. Submission of an executive summary report within 90 days upon completion of the study to SU-IERC.

Prior to commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology and Innovation (NACOSTI) https://oris.nacosti.go.ke and also obtain other clearances needed.

Yours sincerely,

for. Dr Virginia Gichuru,
Secretary; SU-IERC

STRATHMORE UNIVERSITY INSTITUTIONAL ETHICS REVIEW COMMITTEE (SU-IERC)
28 NOV 2019
TEL: +254 (0)703 034 000
P. O. Box 59857 - 00200
NAIROBI - KENYA

Cc: Prof Fred Were,
Chairperson; SU-IERC

APPENDIX D: Research License

REPUBLIC OF KENYA

NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Ref No: **979213**

Date of Issue: **18/March/2020**

RESEARCH LICENSE

This is to Certify that Mr.. Emmanuel Oluoch Olang' of Strathmore University, has been licensed to conduct research in Nairobi on the topic: Sentiment Analysis Model for Detection of Radicalization on Twitter for the period ending : 18/March/2021.

License No: **NACOSTI/P/20/4068**

979213

Applicant Identification Number

Director General
NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Verification QR Code

NOTE: This is a computer generated License. To verify the authenticity of this document, Scan the QR Code using QR scanner application.