Summer 2020

A Study of Information Bots and Knowledge Bots

Amartya Hatua
University of Southern Mississippi

A STUDY OF INFORMATION BOTS AND KNOWLEDGE BOTS

by

Amartya Hatua

A Dissertation
Submitted to the Graduate School,
the College Arts and Sciences
and the School of Computing Sciences and Computer Engineering
at The University of Southern Mississippi
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

Approved by:

Dr. Andrew H. Sung, Committee Chair
Dr. Bikramjit Banerjee
Dr. Ramakalavathi Marapareddy
Dr. Parthapratim Biswas
Dr. Sungwook Lee

August 2020

Published by the Graduate School

ABSTRACT

In this dissertation, a study of different aspects of information bots and knowledge bots is done. The research contributes to a better understanding of the various characteristics of information bots as well as the different patterns and factors responsible for the information diffusion in a social network. This research also shows how these factors can be used to predict information diffusion for a particular topic in a social network.

The second part of the research is focused on strategies for improving the knowledge base of knowledge bots, where two different approaches are studied. In the first approach, knowledge is transferred from other similar types of domains, thus reducing the time and effort required for knowledge acquisition. In the second approach, an attempt is made to generate human-like data, thereby augmenting the knowledge base. To analyze and implement these various methodologies, different machine learning, deep learning and reinforcement learning techniques are used, and encouraging experimental results are presented that demonstrate the great potential of our approaches in applications using knowledge bots.

ACKNOWLEDGMENTS

I would sincerely like to thank my advisor and committee chair Prof. Andrew H. Sung for his continuous guidance, tireless support and excellent mentorship for successful completion of my PhD at the University of Southern Mississippi. I would especially like to thank Prof. Bikramjit Banerjee, who also have a remarkable influence on my academic as a graduate program coordinator and PhD dissertation committee member. Also, I would like to thank other committee members Prof. Ramakalavathi Marapareddy, Prof. Partha Biswas, and Prof. Sungwook Lee. They have provided valuable suggestion and comments for completion of my PhD.

Also, I would like to extend my acknowledgement to the College of Arts and Sciences, all faculties and staffs of the School of Computing Sciences & Computer Engineering for providing me the opportunity to be a student and pursue my graduate study in this institution.

It will not be enough without thanking my beloved parents Mr. Ahsok Hatua and Mrs. Madhabi Hatua for their moral and financial support, love and care and faith and wishes for me. Finally, I would like to thank my friends Mr. Asheshbabu Pothuraju, Mr. Pujan Paudal, Dr. Partha Sengupta, and Mr. Trung T. Nguyen and his family for their generous help during my stay. Finally, I would like to extend my gratitude to my other friends and family members who have always been on my side during my PhD journey.

TABLE OF CONTENTS

ABSTRACT ii

ACKNOWLEDGMENTS iii

LIST OF ILLUSTRATIONS vi

LIST OF TABLES viii

LIST OF ABBREVIATIONS ix

NOTATION AND GLOSSARY ix

1 Introduction **1**
1.1 Bots 4
1.2 Problem and Objective 9

2 Information Diffusion **11**
2.1 Data Collecting and Preprocessing 16
2.2 Information Diffusion Pattern 17
2.3 Prediction Models 21
2.4 Results and Discussion 24
2.5 Applications 40
2.6 Conclusion 41

3 Social Bots on Twitter **43**
3.1 Objective 45
3.2 Datasets Used 46
3.3 Methodology 48
3.4 Results 50
3.5 Towards new dimensions for Exploratory Social Bots Detection 62
3.6 Conclusion 63

4 Knowledge Bots: GO-Chatbot **65**
4.1 Motivation 66
4.2 Dataset 68
4.3 Methodology 69
4.4 Experiments and Results 73
4.5 Discussion 76

4.6 Conclusion 77

5 Knowledge Base Generation Using Generative Adversarial Networks 78
5.1 Methods to Generate Text Data using GAN 80
5.2 Proposed Methods 84
5.3 Result 98
5.4 An Intrinsic Evaluation Metric 101
5.5 Conclusion 104

6 Future Work and Conclusion 106
6.1 Future Work 106
6.2 Conclusion 110

BIBLIOGRAPHY 114

LIST OF ILLUSTRATIONS

Figure

1.1 Things That Happen Every 60 Seconds [7] . . . 1
1.2 Data Information Knowledge Wisdom Model . . . 2
1.3 Transitions from data, to information, to knowledge, and finally to wisdom . . . 3
1.4 Web Crawler . . . 5

2.1 The overview architecture . . . 15
2.2 Sample alignment performed by the DTW algorithm between two series [129] . 18
2.3 Sample dendrogram [56] . . . 19
2.4 Recurrent Neural Network with loop . . . 23
2.5 Different patterns of #tweet . . . 26
2.6 Different patterns of #retweet . . . 27
2.7 Different patterns positive score . . . 28
2.8 Different patterns negative sentiment score . . . 28
2.9 Different patterns neutral score . . . 29
2.10 Different patterns positive percentage . . . 29
2.11 Different patterns negative sentiment percentage . . . 30
2.12 Different patterns neutral percentage . . . 30
2.13 Different patterns of direct influence . . . 31
2.14 Different patterns of indirect influence . . . 32
2.15 The comparison of true value and predictions with LSTM on #tweet dataset . . 35
2.16 The comparison of true value and predictions with LSTM on #retweet dataset . 35
2.17 The comparison of true value and predictions with LSTM on #negative_sentiment dataset . . . 36
2.18 The comparison of true value and predictions with LSTM on #neutral_sentiment dataset . . . 36
2.19 The comparison of true value and predictions with LSTM on #positive_negative dataset . . . 37

3.1 Network Core Members Intersection Plot of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots . . . 51
3.2 K-Core Decomposition Analysis of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots . . . 52
3.3 Robustness Attack Test i) Friendship Network of Traditional Advertisement Bots ii) Friendship Network of Social Advertisement Bots iii) Hashtag Network of Traditional Advertisement Bots iv) HashTag Network of Social Advertisement Bots . . . 53

3.4 Robustness Attack Test i) Mention Network of Traditional Advertisement Bots ii) Mention Network of Social Advertisement Bots iii) RT Network of Traditional Advertisement Bots iv) RT Network of Social Advertisement Bots . 54
3.5 Retweet Diffusion Timeline of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots 55
3.6 Normalized Topic Over Time Distribution of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots 56
3.7 Centrality Leaders Correlation Plot of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots 57
3.8 Content Authoring Homogeneity Chart of: i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots 58

4.1 Diagrammatic representation of three different domains 67
4.2 Diagram of proposed methodology 71
4.3 Tranfer Learning mechanism used in the research 72
4.4 Tranfer Learning mechanism used in the research 72
4.5 Tranfer Learning mechanism used in the research 72
4.6 Performances of TL and TL+AT modles on old datasets 75
4.7 Performance of the proposed model using new datasets 76

5.1 Proposed SAGAN for dialogue generation 85
5.2 Tranfer Learning mechanism used in the research 86
5.3 Tranfer Learning mechanism used in the research 86
5.4 Flowchat of the proposed SAGAN using GCA 89
5.5 Flowchat of the proposed SAGAN using GCA 91
5.6 Encoder of the proposed SAGAN using GCA 91
5.7 Decoder of the proposed SAGAN using GCA 91
5.8 Schematic Diagram of Generator 92
5.9 Distribution of training and result of GCA 100
5.10 Distribution of training and result of SAGAN using GCA 100
5.11 Distribution of training and result of LaTextGAN 101
5.12 Distribution of training and result of SAGAN LaTextGAN 101

LIST OF TABLES

Table

2.1 CVIs for number of clusters 4 to 10 for tweet volume parameter 26
2.2 CVIs for number of clusters 4 to 10 for retweet count parameter 26
2.3 CVIs for number of clusters 4 to 10 for negative sentiment percentage 27
2.4 CVIs for number of clusters 4 to 10 for negative sentiment score 27
2.5 CVIs for number of clusters 4 to 10 for positive sentiment percentage 28
2.6 CVIs for number of clusters 4 to 10 for positive sentiment score 29
2.7 CVIs for number of clusters 4 to 10 for neutral sentiment percentage 30
2.8 CVIs for number of clusters 4 to 10 for neutral sentiment score 31
2.9 CVIs for number of clusters 4 to 10 for direct influenced users 31
2.10 CVIs for number of clusters 4 to 10 for indirect influenced users 32
2.11 Comparison of testing RMSE when using ARIMA and LSTM for different Information Diffusion parameters 34
2.12 Dataset descriptions 37
2.13 List of extracted features 38
2.14 Performance of SVM and RF using extracted features using both datasets . . . 39
2.15 Performance of NLTK and LSTM using extracted features using both datasets . 40

3.1 Statistics about the datasets of categorical bots 60
3.2 Traditional Advertisement Bots 60
3.3 Social Advertisement Bots 61

4.1 RL parameters used for the new dataset 74

5.1 HYPER-PARAMETERS USED FOR THE GCA MODEL AND SAGAN USING GCA MODEL 96
5.2 HYPER-PARAMETERS FOR LATEXTGAN MODEL, SAGAN USING LATEXTGAN MODEL 97
5.3 EXAMPLE OF BLEU SCORE CALCULATION 98
5.4 BLUE SCORE FOR DIFFERENT GAN MODELS 99
5.5 DE SCORE FOR DIFFERENT GAN MODELS 104

LIST OF ABBREVIATIONS

DIKW	-	Data Information Knowledge Wisdom
DTW	-	Dynamic Time Warping
CVI	-	Cluster Validity Index
ARIMA	-	Autoregressive Moving Averag
LSTM	-	Long Short-Term Memory
RNN	-	Recurrent Neural Network
RMSE	-	Root Mean Square Error
NLU	-	Natural Language Understanding
NLG	-	Natural Language Generation
RL	-	Reinforcement Learning
DQN	-	Deep Q-Nets
LU	-	Language Understanding
DM	-	Dialogue Management
DST	-	Dialogue State Tracker
NMT	-	Neural Machine Translation
TL	-	Transfer Learning
AT	-	Attention Mechanism
GO Chatbot	-	Goal-Oriented chatbot
GAN	-	Generative Adversarial Networks
NLP	-	Natural Language Processing
CNN	-	Convolution Neural Network
SeqGAN	-	Sequence Generative Adversarial Networks
MMD	-	Maximum Mean Discrepancy
FM-GAN	-	Feature Mover GAN
EMD	-	Earth-Mover's Distance
GCA	-	Generative Conversational Agents
SAGAN	-	Self-Attention Generative Adversarial Network
BLSTM	-	Bidirectional Long Short Term memory
BOS	-	symbol representing the beginning of the sentence
EOS	-	symbol representing the end of sentence
UNK	-	Unknown Symbol
ROUGE	-	Recall-Oriented Understudy for Gisting Evaluation
BP	-	Bravity Penalty
DE	-	Dialogue Evaluator
LP	-	Length Penalty
BERT	-	Bidirectional Encoder Representations from Transformers

Chapter 1

Introduction

"Over 2.5 quintillion bytes of data are created every single day, and it's only going to grow from there. By 2020, it's estimated that 1.7MB of data will be created every second for every person on earth." – SocialMediaToday

In the present world, data is generated at an exponential rate. According to SocialMediaToday [7], every day, 2.5 quintillion bytes of data is generated. Ninety percent of the total data of the world was produced in the last two years. This enormous amount of data is also called Big Data because of its variety, velocity of generation and volume. This huge volume of data can be used to improve every aspect of our life.

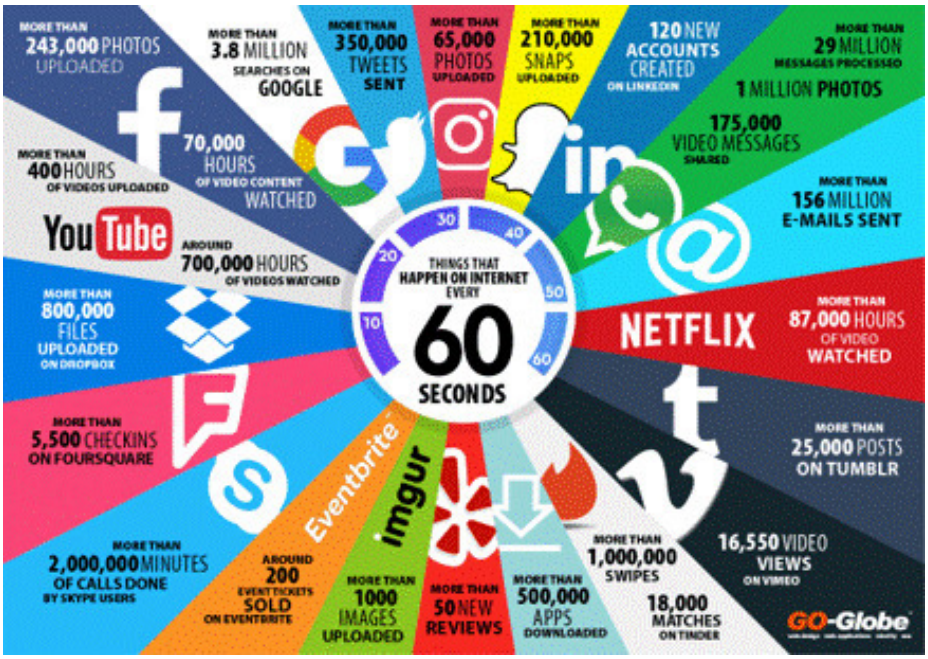


Figure 1.1: Things That Happen Every 60 Seconds [7]

The best utilization of data is possible if the data is appropriately interpreted and comprehended in a specific context. The Data Information Knowledge Wisdom (DIKW) model in Fig. 1.2 represents the hierarchical transformation of data. Data can be interpreted as information and knowledge to get the best benefit out of it. In normal usage, it is not

uncommon for people to use data, information, and knowledge interchangeably, but for us, these have different connotations. The knowledge hierarchy suggests a certain degree of dependence on one another. Data is the prerequisite of information, and information is the prerequisite of knowledge, and of course, knowledge is the prerequisite of wisdom. While this is generally true, it is not necessarily always valid. In this context, we can assume that there is a certain degree of dependence of information on data, and knowledge on information. Data is defined, "As being discrete, objective facts or observations, which are unorganized and unprocessed and therefore have no meaning or value because of lack of context and interpretation" [127]. By its very nature, data is raw facts and figures that we derive from the environment. These facts may relate to some events, entities, or other kinds of transactions. The most important characteristic of data is that it is unorganized until it is processed, and it lacks a context; therefore, it can not be interpreted. The notion of Information can be defined as "organized or structured data, which has been processed in such a way that the information now has relevance for a specific purpose or context, and therefore meaningful, valuable, useful and relevant" [127]. Information is inferred from data; it is processed data, and it always has a meaning and purpose. To illustrate, let us say the sound that one hears is data. If someone infers the source of the sound, that becomes information. In that sense, information is a subset of data. Information is the data that possess context, relevance, and purpose. According to Acoff [22], "Information is found answers to questions that begin with 'who,' 'what,' 'where,' 'when' and 'how many.'"

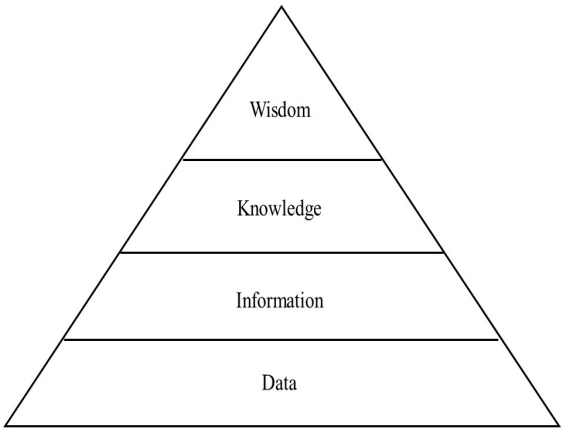


Figure 1.2: Data Information Knowledge Wisdom Model

The third component of the Data Information Knowledge Wisdom (DTKW) Model Fig. 1.2 is 'Knowledge,' which is tough to define. It is typically explained in the context

of information. When we think of knowledge, we are looking at how a particular piece of information is put into use in solving a problem. The context of the problem is essential to understand the notion of knowledge. Knowledge is always acquired through proper study and interpretation of information. For example, if a student studies one course, he or she undergoes a particular process and acquires some knowledge (although this just one of the ways to look at knowledge). This knowledge is obtained based on learning. Knowledge can also be acquired by the process of thinking, observing, and understanding problem areas, among others. So knowledge helps us solve a problem with given information. For this reason, knowledge is also a cognitive process. Knowledge is normally considered as a fluid mix of framed experience, values, contextual information, and expert insight.

There are three types of knowledge: i) procedural ii) declarative iii) semantic or episodic. Procedural knowledge gives the understanding the ability to carry out a particular procedure. Declarative knowledge is a routine knowledge the can readily recall information from short-term memory. Semantic knowledge is very organized and it requires prior knowledge (major concepts, vocabulary, facts, relationships, etc.) and it finds a relation to prior knowledge. Episodic knowledge represents the knowledge of episodes — for example, experimental information. Knowledge can also be classified into two categories: i) tacit knowledge ii) explicit knowledge. Tacit knowledge can be gained from experience, though it is difficult to transfer, while explicit knowledge is well documented and it can be gained from books, reports or documents. There is a theory, however, that documented knowledge is called information.

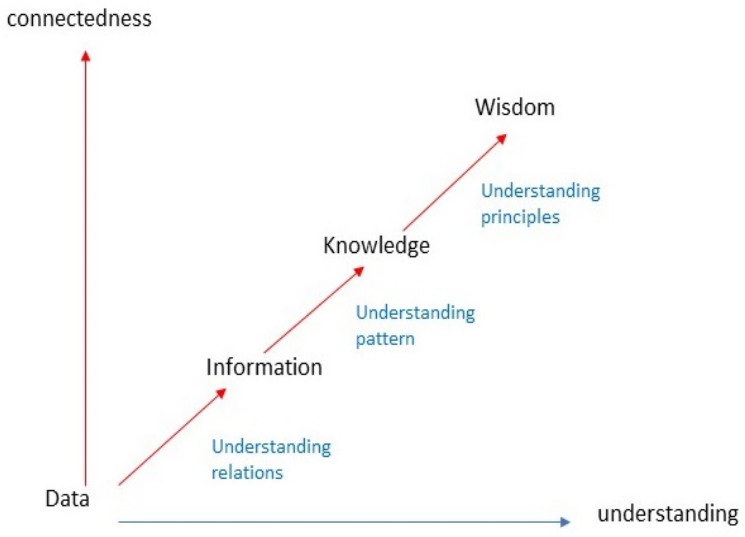


Figure 1.3: Transitions from data, to information, to knowledge, and finally to wisdom

In our practice, we try to capture tacit knowledge by different means and convert it into explicit knowledge. This is called knowledge management. Using knowledge by understanding principles we can gain wisdom. The transition of data to information and information to knowledge is not always very distinct and easily identifiable. Sometimes human interpretation is also needed to contextualize the data to find information and knowledge.

However, understanding, interpreting, and managing that data is not always humanely possible. Instead, automatic computerized programs are used to get the information and knowledge from data. These programs are typically known as bots.

1.1 Bots

Bots are computer programs that perform specific tasks with or without human intervention. Bots are different than regular computer programs as they imitate human behavior, and are always made to appear more human. The bots are typically used for repetitive and continuous jobs. Examples of some of the bots are Web Crawlers, Entertainment bots (Art bots and Game bots), Chatbots, Information bots, Hackers, Spammers, Scrapers, and Impersonators, among others. Bots are mainly used for automating tasks to reduce time and effort. They perform repetitive jobs instead of a human and, more importantly, try to exhibit some human-like characteristics. The significant difference between a typical computer program and a bot is that a bot always tries to impersonate a human and behave like a human so that users do not get a strange feeling interacting with a machine. For example, to integrate two software systems, a project leader needs to follow some predefined steps; in such circumstances, a bot can help to integrate projects and also provide some insights into the project. Bots are becoming more common in the health care industry, not only for some predefined jobs but also for some clinical purposes. The success of Woebot [16] is an excellent example that bots can be used to help people who are struggling with mental health problems. There are eight major categories of bots that exist, such as Chatbots, Web Crawlers, Information bots, Entertainment bots, Hackers, Scrapers, Spammers, and Impersonators. In reality, it has been observed that bots are not always helpful; sometimes bots are developed for evil reasons. Hackers, Scrapers, Spammer, and Impersonators are generally used for not so good purposes.

1.1.1 Web Crawler

Web crawler or spider is a program which automatically traverses a large number of web pages by different hyperlinks and indexes [142]. The web crawler is mainly used in search

engines. Every search engine has its own and unique web crawler. Some of the very famous search engines are: Google Bot for Google, Msnbot/Bingbot for Microsoft, Baidu Spider for Baidu. Although all of them have a different strategy and unique rules for crawling the web, all of them try to achieve some common goals: i) Efficiency ii) Scalability iii) Robustness iv) Extensibility v) Gathering of Quality Content vi) Removal of Duplicate Content vii) Distributed viii) Excluded Content ix) Spam Elimination.

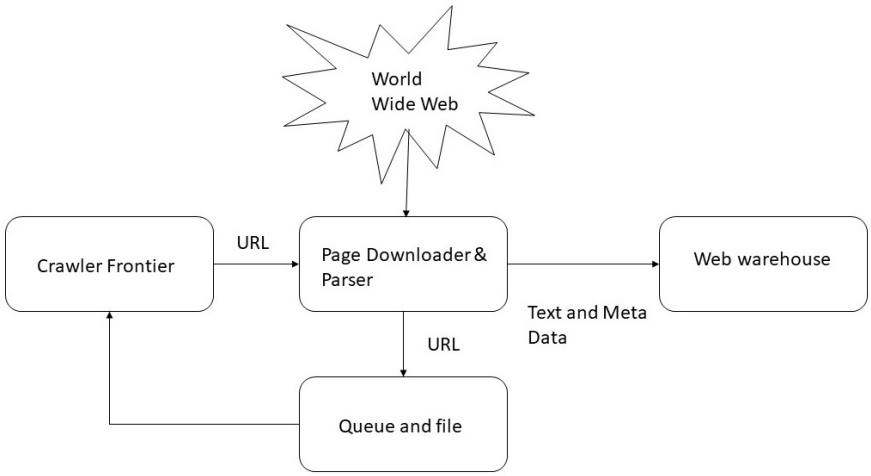


Figure 1.4: Web Crawler

There are eight different types of web crawler present such as: i) Customary Web Crawler ii) Deep Web Crawler iii) Incremental web crawler iv) RIA (Rich Internet Application) Web Crawler v) Unified model - Web Crawler vi) Focused Web Crawler vii) Parallel Crawler viii) Distributed Web Crawler.

1.1.2 Entertainment Bot

Bots are hugely used in the gaming industry which is a major part of the online entertainment industry. In multiplayer games, bots are used as one of the virtual participants on many occasions. Other than gaming online music and video creating apps bots are extensively used as virtual assistance. In many fashion related websites, there are virtual assistants to suggest the user cloths and other important pieces of stuff based on the user profiling analysis.

1.1.3 Hackers

Sometimes bots have been used for some evil purposes. Hacker bot is one example. Trojan bots and key logger bots are very common examples of hacker bots. These bots, which are

normally downloaded from some malicious link or unauthorized software, steal information from the host system. The bots can attack some important areas of a system and can steal passwords and other private information. They also can initiate various fetal activities like a root-kit attack.

1.1.4 Spammers

Spammer bots are used for spamming information on the internet. These bots are normally used for advertisement or other promotional activities. These bots send the same information to the users repetitively through different mediums such as email, text message, suggestions in websites, etc. In the social network, spammers are very popular and used for promotion, advertisement or propaganda. It promotes the same product again and again in the social network and, as per basic human instinct, normal users get attracted to the news or the event that is promoted, especially if it is made visible numerous times. Using social bots to spread rumours, propaganda and particular ideology has become a very common practice on social media sites such as Twitter. Twitter's spambots can tweet about a product or an event many times a day, which contributes to the popularity of the product, event or rumour.

1.1.5 Information Bot

An information bot spreads information. Information bots are normally used to send a piece of information to many people at a time. Broadcast messages in mobile phones or email groups are perfect examples of information bots. The Information spread through an information bot is not always very relevant or equally significant to all the recipients. In our daily life, we receive lots of text messages, email, and social media notifications. The importance of these push messages and notifications changes over time and the current situation in the world around us. For example, these days everyone in the world is very much worried about the pandemic COVID-19 [5] since it is a matter of concern for everyone in the world. Somehow because of this unexpected situation, the 2020 US Election is not getting the highest amount of attention. Depending on the interest surrounding a subject, the receivers of the information forward it to people in their social group (physical and virtual). This is a way the information is transmitted from its source to many receivers. Almost all auto-generated information (such as periodic notifications) are generated by some program or information bot. Different news broadcasting companies actively doing this job and cater a different variety of information to a huge parentage of people in our society. Traditionally, news broadcasting companies perform their tasks mostly though human input; recently, however, there are applications developed where bots are writing and editing news

articles [10]. Even some of the news channels are using robots to read the news articles [11]. Moreover, in the present day where social media is an integral part of our life, social networks are also a prevalent resource for spreading the information.

Social network or social media phenomenon began around 2003. In the initial years of social media, MySpcace and later Orkut were the two most popular social media. Later in 2008 Facebook came joined the list and got lots of popularity. Structurally all of them were very similar and their purpose was to get connected with friends and family. In 2006 Twitter came with a different approach, in Twitter people can follow some event, organization or a person based on their choice and get information directly from the source. This opened up a new horizon of information spreading or broadcasting. Apparently, Twitter may seem like a parallel medium of traditional news media but it lacks the authenticity of the traditional news media. The information on Twitter can be truthful, untruthful, or an opinion of the writer. This is one of the downsides of social media, there is no proper way to judge the correctness of the information. As we discussed before the information can be spread by automated programs or bots, social media also suffers from the problem of fake accounts or impersonators. To identify a real user from fake profile Twitter is providing some verification services to the accounts or user although that is very limited to very popular Twitter users. Moreover, every social media company has its own set of an algorithm to identify any kind of malicious or suspicious activities and stop it immediately. The information spreading over social media is comparable to the spreading of viral diseases in communities. The information spreading or information diffusion models of social media often derived from the models of viral disease spreading models. So characterizing, predicting, and quantifying the impact of postings, tweets, messages, etc. on social media platforms is a topic of growing interest due to the increasing reliance on using social media as a means for various purposes by individuals and organizations alike.

1.1.6 Knowledge Bots

One of the most useful and artificially intelligent bots is knowledge bot. All the above bots can have a different kind of artificial intelligence so all of them can be classified as knowledge bot. The most important characteristic of knowledge bot is it has some definitive purpose and it can take certain decision using artificial intelligence to complete the objective. The more the complex is the objective of a knowledge bot, it is expected to have more analysis power. To fulfil this need all knowledge bots use different machine learning or deep learning techniques to analyze the data, objective and response.

A wide range of applications of knowledge bots is observed in the form of chatbots.

Chatbots can have different mediums of interactions with the users. Chatbots imitate the human conversation process and tries to make an engaging and entertaining conversation. They are also known as Conversational agents. The usage of Chatbots is increasing day by day by various industries to keep in touch with their customers. Hence, it is becoming an integral part of modern business. Banking and Finance, Health care and Pharmaceuticals, Hospitality, Retail, Insurance, Travel and Tourism, Supply chain management, and Logistics are the major industries where Conversational agents are being used extensively. Chatbots can be classified into two categories, transactional bots, and knowledge bots. Transactional bots are simple and work in a minimal domain. Transactional bots can interact in a very restricted fashion. For example, a transactional bot can answer only a set of questions which are given to it, but it can not find a new answer by finding relations between different information. Knowledge bots are generally more intelligent and can deduce different relationships from the knowledge base or database. Knowledge bots use different mediums like text, audio or sometimes video to interact with the users to operate for a wide range of applications. Recently, knowledge bots are getting popularity as personal assistant Amazon Alexa and Siri, Google home are examples such as personal assistant knowledge bots [12].

As per a survey conducted by McKinsey [1], 75% of telecoms respondents are focused on AI for service operations. 59% work on product development for high tech companies and finance companies see 40% focused on risk as their top priority. Addition of bots in the business boots up the collaboration between businesses. On the flip side the experience of users interacting with bots not always very pleasant. Surveys show the sometimes the users faced very extreme experiences. Knowledge bots or conversation agent sometimes asks many questions and not able to identify very unusual situations. This is very common feedback from most of the unsatisfied users. On the other hand, sometimes the users experienced eerie feeling as they are chatting with a machine and it is responding like a human. Hence it opened up a wide range of research areas for artificial intelligence (AI), psychology, cognitive science etc.

Up to this point, it can be observed that the discussion started with the new paradigm of Big Data and its characteristics. Moving forward to that we have seen the importance of Data, Information, Knowledge, Wisdom (DIKW) model and also realized that to get the benefit from the Big Data DIKW model is required. The volume, velocity and variety of Big Data need very fast computation, hence different types of bots are needed to get information, knowledge and wisdom from Big Data. So we discussed different types of bots and their applications. Now if we try to connect the topics have been discussed till now, we can get a clear picture where the Big Data can be used to get benefits in various aspects of our life using different types of bots to convert data to information, knowledge or

wisdom depending on the necessity. Analysis and interpretation of Big Data using different automated programs and bots plays a major role in the successful application of DKIW model and getting the benefit of it. The success of a bot also depends on many factors and as the complexity of information and knowledge gets higher the bots also use very advanced and complex algorithms.

In this research work, the DIKW model is explored using information and knowledge bots. To pursue this research work social media and several freely available data source are also used as a source of data. This data is being spread over different social media in the form of information which satisfies the second component of the DIKW model. Information spreading (or information discussion) pattern and factors behind these patterns is a major part of this study. These factors lead to the fact that all the users of social networks are not always human. So a section of this study also devoted to identifying the characteristics and patterns of bots and social bots on the social network. As discussed previously, all applications of knowledge bots are increasing day by day in our daily life, the same thing can be observed in social media as well. Different social media users (different companies) are using knowledge bots/ auto replying agents/ conversational agents/ chatbots with their social media platform to get in touch with the customer always. This is the next part of this study which is the third part of the DIKW model. So the last part of this study focuses on the knowledge bots. This research focuses on data, information and knowledge only as we go up in this hierarchical model the difference between two classes becomes more blur. Hence, this research is restricted until identifying knowledge from information. The elaborate discussion on these topics are provided in the following chapters. Chapter [2] and [3] is dedicated for information bot and Chapter [4] and [5] finally conclusion and future work is discussed in Chapter [6].

1.2 Problem and Objective

1.2.1 Problem

The research of Information and Knowledge bots is an attempt to understand the different characteristics of these two types of bots and how they are dealing with the contexts of information and knowledge about various subjects. As we go from the bottom to the top in the Data, Information, Knowledge model, more intelligence is required to differentiate between data and information and knowledge. We can then say that data alone have no value unless there is a context added. This makes the data information, and information is more beneficial when it is analyzed and interpreted with different intelligent methods and projected it as knowledge. The problem of with research can be divided into two

parts. Understand the characteristics of different types of Information bots and patterns of information diffusion is the first part of the problem. The second part of the problem is to identify the methods to enhance the understanding of the information intelligently so that the Knowledge bots can answer all the questions they face.

1.2.2 Objective

This research is mainly focused on Information and Knowledge bots. Information and Knowledge bots cover a wide range of applications. To use publicly available data and a dynamic network, Twitter [14] is used as the domain for Information bots research. Since Twitter is a large and dynamic online social network that explores the pattern of interaction between users and Information bots, understanding the structure of the network is crucial. Secondly, understanding the model of information diffusion on Twitter to identify the important parameters and their interaction is also very important to study. Hence, the main objectives of studying Information bots are:

A. To analyze and understand the network structure and interaction pattern between users on Twitter.

B. To analyze and identify different kinds of information diffused on Twitter, and how factors such as influence, volume, and sentiment of Tweets are involved in their diffusion.

On the other hand, chatbots are used for the study of Knowledge bots. Knowledge base plays the most critical role in developing a Goal-Oriented (GO) chatbot. A GO chatbot is as good as its knowledge base. Most of the time the quality of knowledge base suffers because of inadequate data. The chatbot study is focused on two essential aspects. Firstly, how to use similar types of data from a different domain and solve the problem of data inadequacy. Secondly, how to generate data that is very similar to human-generated data. Hence the two objectives of Knowledgeable bot research are:

A. To develop a goal-oriented chatbot for domains with insufficient data by using similar types of available data from other domains.

B. To improve the performance of chatbot and to develop a knowledge base.

Chapter 2

Information Diffusion

Since the inception of online social media, its usage has evolved in many facets, and nowadays social networks have become one of the most important means of communication: they are used as a major tool for viral marketing, political messaging, opinion formation and many other things. The abundance of information in the social network and its huge impacts made social networking a fast growing industry in recent years. People who share a common interest get connected over the social network; they share and spread information over a period of time, or, in other words, the information diffuses over the social network. Information diffusion is an important function of online social networks, which are based on activities like viral marketing, advertisement, election campaigning, information propaganda and others. The pattern and behavior of information diffusion in the social network, therefore, can be analyzed to help predict the success, failure, popularity and opinion about different events.

In the previous research works on information diffusion on the social network, Mark Granovetter [60] used a graph-based method, and this work was very popular as graph-based methods are the earliest form of the information diffusion model. The information diffusion pattern is also compared with the viral diseases spreading patterns by Lars et al. [20]. Initially, these methods were effective for a small and stable network. The modern social media platforms (like Twitter) are massive in structure and very dynamic, where the popularity of a topic or content (meme or hashtag) depends on several factors. The primary reason for a topic to become famous is the popularity of the content producer or the number of friends or acquaintances of the content producer [20]. Later it has been found that the popularity of the content producer is not the only reason behind a popular hashtag or viral content. Further research shows that information diffusion a very complex behavior and there are many factors involved to guide pattern of information diffusion such as: the sentiment of the tweet, topic of the tweet, the network and community structure of the user, the physical location of user and involvement of some popular or influential users on a topic. In Twitter, the information is diffused by hashtags or memes. The users normally express their opinions about a certain topic and, based on the mood or sentiment of the discussion, the rate of diffusion changes over time. The complex and dynamic relation

between information diffusion on Twitter and sentiment of tweets are analyzed and studied by Ferrara in [54]. In [54], Ferrara focused on, i)the relation between sentiment and rate of information diffusion and content popularity, ii) temporal evolution of different types of sentiment. For example, sad news spreads much faster than good news at first, while good news maintains a steady spreading rate. The research work of Ferrara [54] is one of the important references of this research. However, he focused only on the sentiment of tweets as the reason behind the information diffusion on Twitter. He did not discuss the topic of tweet or influence of the content creator. Later Pinto et al. [121] have addressed these two-issues while proposing their information diffusion model in the social network. In [121], Pinto et al. proposed a framework to model information diffusion based on linear multivariate Hawkes processes and for topic modeling they used Latent Dirichlet Allocation (LDA) topic model [25]. The primary focus of this model was information diffusion on Twitter and the trending topics on social media. Although topic modeling is a very important strategy in understanding information diffusion processes in social networks, it does not provide enough information unless we understand the mood or sentiment of the information. Sometimes the mood of information or tweets are clearly understandable from their topic, though that is not always true. So, we need to identify the sentiment of the information to understand the information diffusion model for social media.

Up to this point of discussion we have seen, sentiment and topic of tweets are critical factors of information diffusion for new social media network. Other than the content of the tweets, the structure of the network is also significant. The dynamics between information diffusion and the different participants of the social network is discussed by Yang et al. in [155]. In their research, they proposed an information diffusion model in implicit networks. To implement this model, they used Linear Influence Model (LIM) this another important reference of the current research. In social media, an influential person will be connected to many users or followers. If we consider the graph like the structure of a social network, then the degree of a vertex is higher if that person is trendy in social media. So the information created by a popular Twitter user will spread to a massive number of the Twitter user very quick. In a big and dynamic network, this can be considered as of the parameters responsible for different information pattern.

In [125] Reagans et al. discuss how network structure plays a vital role in information diffusion; the authors conclude that social cohesion and network range are more important than the strength of the tie between two people for effective knowledge transfer, and information diffusion. In further research, Kafeza et al. [80] proposed an information diffusion model for Twitter. In their research, they focused on a particular hashtag and collected data related to that hashtag for a long time. By analyzing the collected data,

they proposed different Tree-Shaped Tweet Cascades patterns of information diffusion on Twitter. For generalization, they identified the four most popular tree structure; those are representing information diffusion patterns on Twitter. They also proposed an information diffusion pattern prediction model where linguistic features, user profile's information and their tree-based tweet patterns are used as the important features. This is also a significant research work which influenced the current research work. In [80], the researchers used a tiny dataset, so while extending this research in the present context, we created a big dataset to get a better understanding of the information diffusion pattern. Moreover, in the present research work, the data is analyzed, and the patter is identified using some unsupervised machine learning algorithms such as clustering using Dynamic Time Wrapping (DTW) method. It has been observed that in most of the earlier works, the focus of the research was to identify the pattern of information diffusion on the social network. In contrast, minimal effort has been given to predicting the pattern. This research is aimed to propose a model which understand the pattern of information diffusion and forecast its future trend also.

Identifying the sentiment of tweets is an extremely challenging problem. Tweets are usually short texts and therefore it is difficult to analyze their sentiment. In Twitter, different features can be identified such as "tweet," "retweet," "reply," "direct message," and "like." There are several existing types of research on the sentiment analysis of tweets. Richard Colbaugh and Kristin Glass [38] proposed a model which collects corpora from Twitter to performing opinion mining and sentiment analysis. Most of the sentiment detection algorithms are designed to identify user opinions about products rather than user behaviors. Mike Thelwal [138] implemented the SentiStrength algorithm to determine the strength of sentiments from an informal text. Nasir Naveed [110] proposed a content-based analysis of interestingness on Twitter using LDA and regression methods to show how tweets containing negative sentiment travel faster when compared to positive sentiment tweets. Much research has been done on sentiment analysis of tweets and retweets. Retweet count may convey the popularity of a topic, but to understand the sentiment associated with the topic or hashtag, it is necessary to analyze the replies. Here is an example tweet:
Tweet: Fireworks on July 4th
Reply 1: Its an eye feast
Reply 2: Never like before, awesome
Reply 3: Suffered with smog
Reply 4: Nashville is hosting USAs biggest event
Reply 5: Hope the anniversary would be marked for years to come by "guns" and "bonfires" and "illuminations."

In the above-mentioned tweet, the tweet text alone is not expressing any positive sentiment or negative sentiment. So, to understand the sentiment or mood of a tweet or a conversation in Twitter, not only its topic but the context and mood of other users is also very important.

This research is aimed to build a prediction model, which can predict or forecast the volume of tweets or in other words the popularity of the information for a particular topic which is represented by a hashtag. This model also predicts the reaction of all the users who receive the information and total number of people are influenced by the information or reachability [38] of the information. This prediction can be done in a time bound manner, which can be represented by a time series model. The proposed model predicts three different facets of information diffusion: i) volume ii) sentiment and iii) influence of different popular memes of a social network. In this whole research Twitter is used as the social media platform because of its huge and dynamic nature, freely available datasets by APIs. The objectives of this research work are mentioned below:

1) Understanding the pattern of information diffusion related to a hashtag and its relationship with the volume of tweets and number of people who are using that hashtag. After that, predicting the number of tweets and people who will use the same hashtag over a period of time.
2) Finding the relation between the sentiment of a tweet and its effect on information diffusion and predicting the sentiment of tweets related to a hashtag.
3) Finding both the directly- and indirectly-affected users by a hashtag and predicting the number of total affected users by a particular hashtag over a period of time.

In the following sections methodology, data collection and preprocessing, information diffusion pattern recognition, information diffusion model prediction, experiment setup, results and discussions are described respectively. This research is published in [69].

This section will explain our methodology to modeling the information diffusion process on an online social network, which is Twitter in the scope of this research.

2.0.1 Modeling the Information Diffusion Process

While going through the previous works related to information diffusion, it has been observed that the information diffusion model often designed using the detailed knowledge of the social network or the vast and dynamic nature of the social network is minimized into a generalized form. In this research, we model the information diffusion process on a social network as a multivariate time series problem.

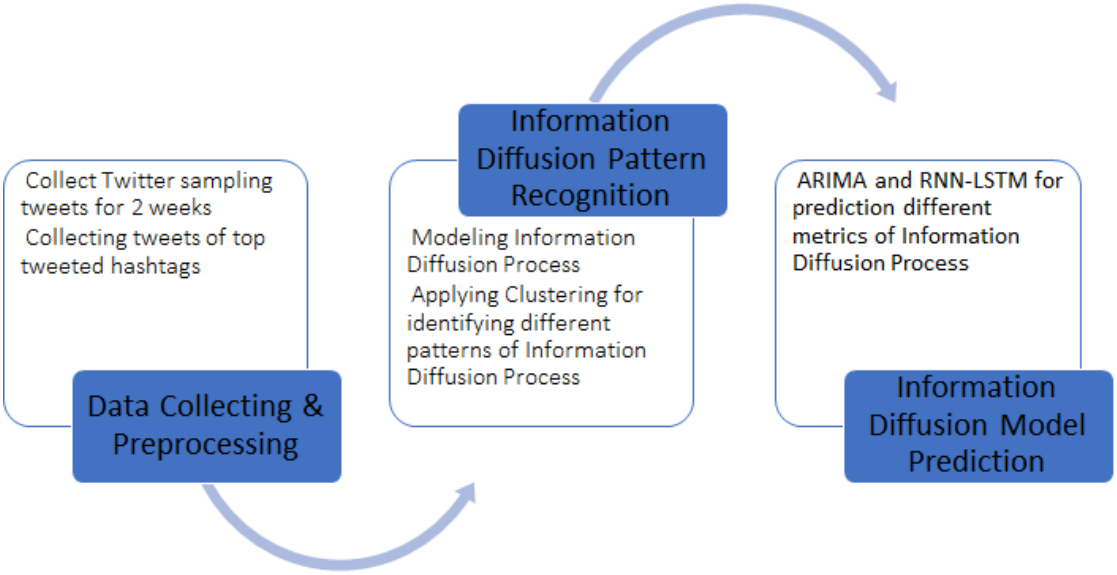


Figure 2.1: The overview architecture

This method is chosen because it does not require the explicit knowledge of the social network. The proposed information diffusion model consists of three time series model with a total of 10 different features. The description of each of the time series model or the dimension and related features are mentioned below:

A Volume: It is the first dimension, and it consists of two features:

- #tweet: the total number of tweets related to a hashtag.
- #retweet: the total number of retweets related to a hashtag.

B Influence: It is the second dimension, and it consists of two features:

- #direct_influence_user: the total number of users and mentioned users related to a hashtag
- #indirect_influence_user: the total number of followers of all users and mentioned users that related to a hashtag.

C Sentiment: It is the third dimension, and it consists of six features:

- #positive percentage: the percentage of positive sentiment among the tweets.
- #neutral percentage: the percentage of neutral sentiment among the tweets.
- #negative percentage: the percentage of negative sentiment tweets.

- #positive average score: the average score of positive sentiment tweets.
- #neutral average score: the average score of neutral sentiment tweets
- #negative average score: the average score of negative sentiment tweets.

2.1 Data Collecting and Preprocessing

2.1.1 Twitter Data Collecting

As analyzing information diffusion on Twitter is the first major step of this research and the information is diffusing based on hashtags on Twitter. So hashtags have a very crucial role in this research. Twitter users normally tweets something and mention a hashtag related to the topic of the tweet. The addition of hashtag with every tweet has created a global information transmission effect on Twitter because hashtags help users keep track of information topics and therefore, can form dynamic communities or groups. To identify the most popular or trending hashtags, we collected Twitter's streaming data for two weeks, from 01-July-2017 to 14-July-2017 using Tweepy Python library. From this initial dataset, all the hashtags are identified with their corresponding tweets are counted. At this point, we collected more than 1 million hashtags and their corresponding tweets. From this 1 million hashtags top 1,686 hashtags are segregated because at least 200 tweets were present for each of the hashtag. According to Twitter API, streaming API will only return 1% of real-time tweet data at a time, so we think that we may not have enough tweets of those 1,686 hashtags to analyze. Therefore, we began to collect all tweets that related to those 1,686 hashtags using Twitter Search API in the next three weeks from 15-Jul-2017 to 04-Aug-2017. Finally, we collected about 27.5 million of tweets that contained those 1,686 hashtags that we wanted to analyze.

2.1.2 Data Preprocessing

Once the data collection process is over, the next step is data preprocessing. Twitter data comes with lots of information. For this experiment, we needed information related to the tweets, data time and user's followers-followee count. All this relevant information is extracted from the 27 million tweets. The count of each of the ten features is done hourly basis as this data is finally used in a time series model.

- #tweet: The total number of tweets that contain such hashtag in each hour.
- #retweet: The total number of retweets that contain such hashtag in each hour.
- #direct_influence_user: The total number of users and mentioned users that associated with all tweets contain such hashtag in each hour.

- #indirect_influence_user: The total number of followers of all users and mentioned users that associated with all tweets contain such hashtag in each hour.
- #positive percentage: The percentage of positive sentiment tweets in each hour.
- #neutral percentage: The percentage of neutral sentiment tweets in each hour.
- #negative percentage: The percentage of negative sentiment tweets in each hour.
- #positive average score: The average score of positive sentiment tweets in each hour.
- #neutral average score: The average score of neutral sentiment tweets in each hour.
- #negative average score: The average score of negative sentiment tweets in each hour.

The sentiment of tweets is characterized by six different parameters. The sentiment analysis is done using Python NLTK library and Wordnet corpora [147]. The preprocessed data is available in [9].

2.2 Information Diffusion Pattern

2.2.1 Motivation

After data collection and data preprocessing the next step in the pipeline is analysis of the data and identifying different information diffusion patterns. The data used in this experiment is not labeled so there were be many unknown patterns. In other words there is no label or class name corresponding with each hashtag. To divide hashtags into groups of similar patterns, many time series clustering techniques are employed. As our information diffusion contains ten features, clustering for each of those features is done separately.

2.2.2 Time Series Distance Measure

In the present scenario, our dataset contains multiple sequences that were taken at successive, equally spaced points in time; this is similar to other time series data like stock market data or weather data. To measure the similarities between two temporal sequences which may vary in speed, the Dynamic Time Warping (DTW) distance is one of the most popular measures. Hence DTW is used in conjunction with time series clustering techniques in our experiments.

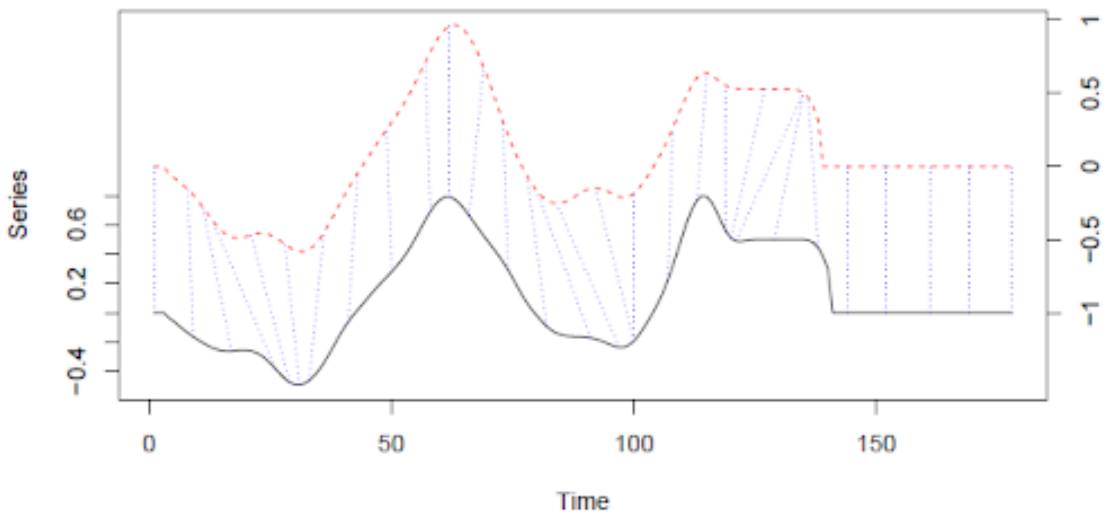


Figure 2.2: Sample alignment performed by the DTW algorithm between two series [129]

2.2.3 Dynamic Time Warping (DTW)

The distance is the most useful parameter in time series analysis which helps to measure the dissimilarity between two time series data. DTW employs a dynamic programming algorithm to find the distance between two time series; though an effective distance measure, it requires a lot of computation. In Fig. 2.2 alignment between two sample time series are shown. The dashed blue lines exemplify how some points are mapped to each other, which shows how they can be warped in time. Note that the vertical position of each series was artificially altered for visualization. DTW requires that the initial and final points of the series match.

To compute DTW [129] the following steps are needed to be followed: suppose x and y are two time series and n and m are their length respectively.

Step1: Create local cost matrix (LCM). The matrix contains distance between every pair of points from each of the sequence. So, the dimension of the matrix is $n \times m$. The distance is l_p norm between any pair of points which is calculated by Equation 2.1.

$$lcm(i,j) = (\sum_{v} |x_i^v - y_j^v|)^{\frac{1}{p}} \tag{2.1}$$

Step 2: After computing the LCM, the next step is to find the path that minimizes the alignment between x and y. Suppose $\phi = (0,0),..,(n,m)$ be the optimum path containing all the points, then the final distance can be measured by Equation 2.2.

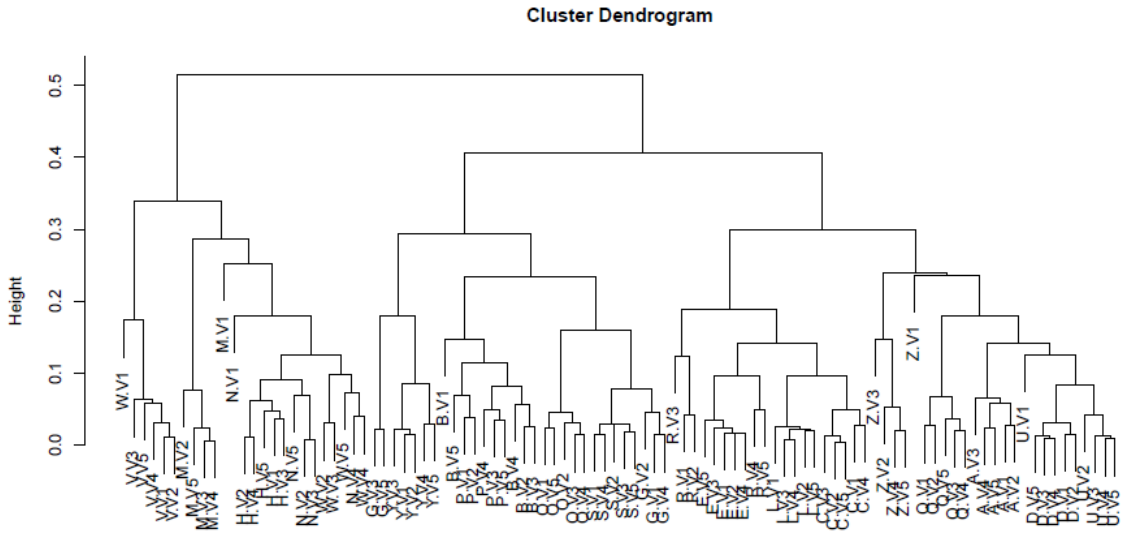


Figure 2.3: Sample dendrogram [56]

$$DYW_p(x,y) = (\sum \frac{m_\phi lcm(k)^v}{M_\phi})^{\frac{1}{v}}, \forall k \in \phi \tag{2.2}$$

DTW distance is typically used jointly with clustering algorithms on time series data. Some of the well-known clustering techniques are Hierarchical clustering, Partitional clustering, and TADPole clustering. Brief descriptions of those clustering algorithms are described in the following subsections.

2.2.4 Hierarchical Clustering

This clustering algorithm tries to create a hierarchy of groups in which, as the level in the hierarchy increases, clusters are created by merging the clusters from the next lower level, such that an ordered sequence of groupings is obtained [56]. The created hierarchy can be visualized as a binary tree where the height of each node is proportional to the value of the intergroup dissimilarity between its two daughter nodes.

2.2.5 Partitional Clustering

Partitional clustering follows a stochastic procedure that begins with a fixed number of random points from its dataset. The number of data points is decided by the required number of clusters. Some of the most popular algorithms of this type are k-means [120] and k-medoids [67]. In the first step of this algorithm a fixed number of data points is randomly selected (say k points) and assigned as centroids. In subsequent steps all the remaining data

points are clustered, one by one, based on similarity to the centroids, and after each iteration new centroids are calculated.

2.2.6 TADPole Clustering

TADPole Clustering is a relatively new method for time series data clustering with DTW distance. In this algorithm, the centroid of the clusters is always the element of dataset, so it can be also considered as PAM clustering. Depending on cutoff value of distance the clustering algorithm is deterministic in nature. To find close neighbors in DTW space, the algorithm initially uses the upper and lower bounds of the DTW distance. To do a faster calculation of clustering, the algorithm tries to prune as many DTW calculations as possible.

2.2.7 Fuzzy Clustering

In previously discussed clustering techniques, each data point of dataset belongs to at most one cluster after the clustering process. However, in the fuzzy clustering technique, each data point of the dataset belongs to each cluster to a certain percentage or degree. If we add the degree of belongingness of a data point across all clusters, then the sum will be one. If N is the number of data points in the dataset and k is the desired number of clusters, a membership matrix u can be created with dimension $N \times k$, where all the rows must sum to one.

In a Fuzzy clustering category, Fuzzy c-means is one of the most popular algorithms. Fuzzy c-means a clustering algorithm tries to create a fuzzy partition by minimizing the function in Equation 2.3a, under the constraints given in Equation 2.3b. The centroid function used by Fuzzy c-means calculates the mean for each point across all members in the data, weighted by their degree of belonging.

$$\min \sum_{p=1}^{N} \sum_{c=1}^{k} u_{p,c}^{m} d_{p,c}^{2} \tag{2.3a}$$

$$\sum_{c=1}^{k} u_{p,c} = 1, u_{p,c} \geq 0 \tag{2.3b}$$

2.2.8 Cluster Evaluation

Clustering is an unsupervised procedure, so performance evaluation of clustering may be somewhat subjective. Much research has been done to develop a cluster evaluation metrics by cluster validity indices (CVIs), and there are many indices proposed by different researchers (might benefit you to list some of these researchers here). In this paper, we

employ with some of the very popular [18] indices among them. For some indices, the higher the value then the better quality of cluster; on the other hand, some indices show exactly the opposite characteristics. Some indices do not concern how the clustering is happening internally, or how the partition works. For example, Silhouette index is an internal CVI and Variation of Information [105] is an external CVI. Time-Series Clustering Algorithms mainly represent a group of different types of clustering algorithms such as Hierarchical clustering, Partitional clustering, TADPole clustering, and Fuzzy clustering. In our experiments, TADPole clustering gave the best results among all the clustering algorithms.

2.3 Prediction Models

As described in section 2 above, we model information diffusion process on Twitter as multivariate time series problem. Time series analysis is one of the difficult problems in Data Science and is still an active research interest area. There are many time series data examples around us, such as predicting stock prices, energy prices, sales forecasts or energy consuming loads (among others). The stochastic nature of these events makes time series forecasting a very difficult problem.

Traditional Time Series analysis follow parametric methodology by decomposing the data into many components such as trend, seasonal and noise components [28]. Techniques such as auto regression, moving average, and ARIMA (p,d,q), etc. are used to analyze time series. However, because of the ability of capturing complex structures of time series models, stateful RNNs such as LSTM are found to be very effective in time series analysis recently.

2.3.1 ARIMA Model

ARIMA stands for Autoregressive Integrated Moving Average. It is a class of model that can capture the temporal structure with different cyclicity in a time series data [29]. ARIMA is most generally used for time series data which can be made to be stationary by differencing (if necessary). A random variable that is a time series is stationary if all its statistical properties are constant over time. A random variable of this form can be viewed as a combination of a signal and noise. The series wiggles around the mean with constant amplitude. ARIMA is a generalization of a simpler Autoregressive Moving Average (ARMA) model that adds the notion of integration. This acronym is descriptive, capturing the aspects of the model itself [19]:

- AR: Autoregression. The component of the model to forecast the interest variable using linear combinations of past values of that variable.
- I: Integrated. The use of differencing steps (i.e. subtracting values at current time step from values at the previous time steps) to make the time series stationary.
- MA: Moving Average. Another component of the model that uses past residual errors in a regression-like model to forecast.

ARIMA (p,d,q) model [19] has three parameters p,d,q:
- p: number of autoregressive terms or the lag order.
- d: the number of non-seasonal differences needed for stationary
- q: the size of the moving average window

In terms of y, the general forecasting equation is:

$$(1-\sum_{i=1}^{p}\phi_i L^i)(1-L)^d Y_t = (1+\sum_{i=1}^{q}\theta_i L^i)\varepsilon_t \tag{2.4}$$

In Equation 2.4, L is the lag operator which operates on a value of a time series to produce the previous value, ϕ_i are the parameters of the autoregressive, and θ_i are the moving average parameters, following the convention introduced by Box and Jenkins [19]. To identify the appropriate ARIMA model for Y, firstly the order of differencing (d) needing to stationarize the series must be determined. Later, the gross features of seasonality characteristics in time series Y are removed in conjunction with a variance-stabilizing transformation (logging or deflating). After above steps, the differenced series can merely fit a random walk or random trend model. However, this stationarized series may still have autocorrelated errors. Therefore, some number of AR terms $(p \geq 1)$ and/or some number MA terms $(q \geq 1)$ are also required in the final predicting model.

2.3.2 Recurrent Neural Network (RNN)

Recurrent Neural Networks (RNN) [18] are a class of neural network which are well suited for sequential data. This makes them a compelling model for time series, forecasting tasks etc. RNN can be built in many ways. One of the simplest ways to understand RNNs is to think of them as a feed forward neural network that has been unfolded in time. Fig. 2.4 below describe the process of unfolding visually in a RNN. At each time step, the network emits an intermediate output o_t and maintain an internal state $s_t, x_t's$ form the sequential input being fed to the network.The following equations describe the update equations:

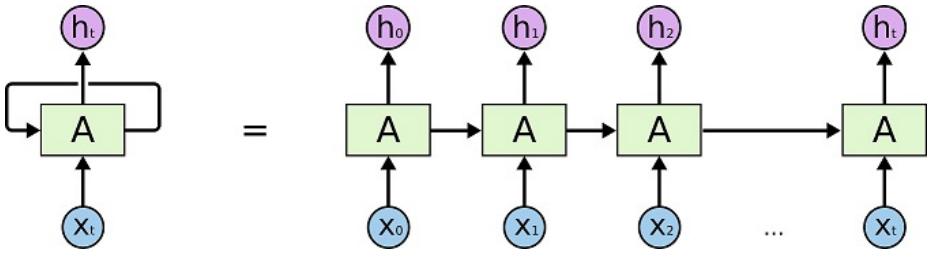


Figure 2.4: Recurrent Neural Network with loop

$$a_t = b + W s_{t-1} + U x_t$$

$$s_t = tanh(a_t)$$

$$o_t = c + V s_t$$

$$y_t = softmax(o_t)$$

The matrices U, V and W form the parameters of the model which are learnt by standard propagation. In practice, RNNs have limited usefulness because they suffer from the problem of vanishing and exploding gradients.

The vanishing gradient problem occurs when the gradient values become zero and the exploding gradient problem occurs when the gradient values blow up to infinity. Without going into the mathematical details, the reason why this happens is as follows. Because of the chain rule of differentiation, during the propagation step, gradients at each time step are multiplied together. If this value is less than one, the successive multiplications will drive this value to zero. If this value is greater than one, successive multiplications drive this value to infinity.

2.3.3 LSTM for Time Series Prediction

Long Short-Term Memory (LSTM) [72] is a type of recurrent neural network which protects gradients from harmful changes during training and can capture dependencies when there are time lags of unknown size. LSTM can remove or add information to the cell state by regulated gates. The key to this ability is that there is no activation function within the recurrent components. Thus, the stored value, is not iteratively squashed over time and the gradient term does not tend to vanish or explode when backpropagation through time is applied to it. Fig. 2.4 shows the internal gates and connections of a standard LSTM cell. The following equations describe the update equations of LSTM model:

$$i_t = g(W_{xi} x_t + W_{hi} h_{t-1} + b_i)$$

$$f_t = g(W_{xi}x_t + W_{hi}h_{t-1} + b_f)$$

$$o_t = g(W_{xi}x_t + W_{hi}h_{t-1} + b_o)$$

$$c\_in_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

$$c_t = f_t c_{t-1} + i_t c\_in_t$$

$$h_t = o_t tanh(c_t)$$

The variables i_t, f_t, o_t are the input, forget and output gates respectively. The gates values can be reset either after feeding each batch or after feeding the entire sequence.

2.4 Results and Discussion

2.4.1 Data Descriptions

Once preprocessing step is done, the data is ready to use for the pattern identification and prediction phase. The dataset is having ten time series subsets that corresponds to three dimensions in our information diffusion model: Volume dimension (#tweet and, #retweet), Network Influence dimension (#direct influence user, #indirect influence user), and Sentiment dimension (#positive sentiment percentage, #neutral sentiment percentage, #negative sentiment percentage, #average positive sentiment score, #average neutral sentiment score, and #average negative sentiment score). Each subset of those time series data contains 1,687 samples with 467 measured time steps in an hourly basis.

2.4.2 Time Series Clustering

In clustering operations, the prior decision about the number of clusters carries a lot of importance in obtaining satisfactory results. In this case we performed ten different experiments to do clustering, and the number of clusters is changed every time. So, these experiments are performed for cluster numbers between 4 and 10. After the clustering, standard cluster validity indices (CVIs) are used to determine the best cluster number between 4 and 10. In this case, we have used internal CVIs for cluster evaluation. Internal CVIs and their optimization conditions are mentioned below:

- Sil: Silhouette index [84] to be maximized to get better cluster.
- D: Dunn index [84] to be maximized to get better cluster.
- COP: COP index [84] to be minimized to get better cluster.
- DB: Davies-Bouldin index [84] to be minimized to get better cluster.

- DBstar: Modified Davies-Bouldin index [128] to be minimized to get better cluster.
- CH: Calinski-Harabasz index [84] to be maximized to get better cluster.
- SF: Score Function [128] to be maximized to get better cluster.

In the present context, clustering is performed on the preprocessed Twitter data. The data mainly represents three different dimensions: tweet and retweet count for every hashtag; positive, negative, and neutral sentiment score and percentage of tweets on those hashtags; and the influence measurements of each hashtag. In our experiments, we applied time series clustering with the number of clustering parameters set from four to ten and compared their CVIs. These workloads were considered as one job. Therefore, a total of ten jobs are performed for each of the selected parameters' value. Two jobs were performed for the tweet and retweet volume for every hashtag, six jobs were performed for the positive, negative, and neutral sentiment score and percentage of tweets on those hashtags, and finally two jobs were performed for the direct and indirect influence count of each hashtag. While performing the experiments, it was observed that the best CVIs indices are dependent on the parameter of the selected number of clusters. As a result, the optimal number of clusters that can help gaining maximum best values of CVI indices are recorded.

Clustering for Tweet and Retweet Volume for Every Hashtag:

To find out the optimal number of clusters for tweet and retweet volume features, the different CVIs of different number of clusters from four to ten are compared. Based on the comparison, the best number of clusters for tweet and retweet volume is six. Hence, both tweet and retweet volume data are clustered into six clusters. While in Table 2.1, different CVIs for cluster numbers four to ten are displayed for tweet volume, different CVIs for cluster numbers four to ten are displayed for retweet volume in Table 2.2. In these two tables, the column name represents the test value of the number of cluster parameters (K). Fig. 2.5 and Fig. 2.6 are the pattern visualizations of all six different clusters for tweet and retweet volume. In these figures, the x-axis is representing the time in hours and the y-axis is representing the count of tweets in each hour.

Clustering for Different Sentiment Scores and Percentages for Every Hashtag:

Similar to tweet and retweet volume features, cluster analysis has also performed for different parameters of sentiments of tweets. Every tweet is given three different sentiments: positive, negative and neutral. Each of the sentiments has two different measures: first is the average sentiment score and second is sentiment percentage, which ranges from 0 to 1. The best

Table 2.1: CVIs for number of clusters 4 to 10 for tweet volume parameter

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	0.007	0.004	0.068	0.024	0.041	0.048	-0.004
SF	0.005	0.0004	0.011	0.006	0.009	0.004	0.004
CH	62.15	46.91	193.24	102.76	0.013	0.019	0.017
DB	2.56	6.43	1.88	8.44	45.48	11.88	2.32
DBstar	7.70	2.58	2.37	9.97	15.02	20.37	32.83
D	0	0	0	0	0	0	0
COP	19.02	61.02	8.68	60.88	43.89	66.86	45.85

Table 2.2: CVIs for number of clusters 4 to 10 for retweet count parameter

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	-0.1413	-0.1315	-0.0953	-0.1366	-0.0994	-0.1283	-0.1275
SF	0.0410	0.0189	0.0017	0.0036	0.0038	0.0098	0.0007
CH	39.0036	38.9829	55.7518	52.7244	56.3037	33.9213	124.6780
DB	1.6294	1.5824	1.5051	1.5907	1.5201	1.5301	1.7140
DBstar	1.9392	1.9640	1.8867	1.9991	2.8909	2.2354	2.9524
D	0.0260	0.0260	0.0020	0.0105	0.0105	0.0177	0.0119
COP	1.0800	1.0656	0.8991	1.0337	0.9104	1.0603	0.8111

number of clusters for all the sentiment features is six. While Table 2.3 shows different CVIs values for negative sentiment percentage features, Table 2.4 displays different CVIs values for average negative sentiment score features. Similarly, while Table 2.5 shows different CVIs values for positive sentiment percentage features, Table 2.6 displays different CVIs

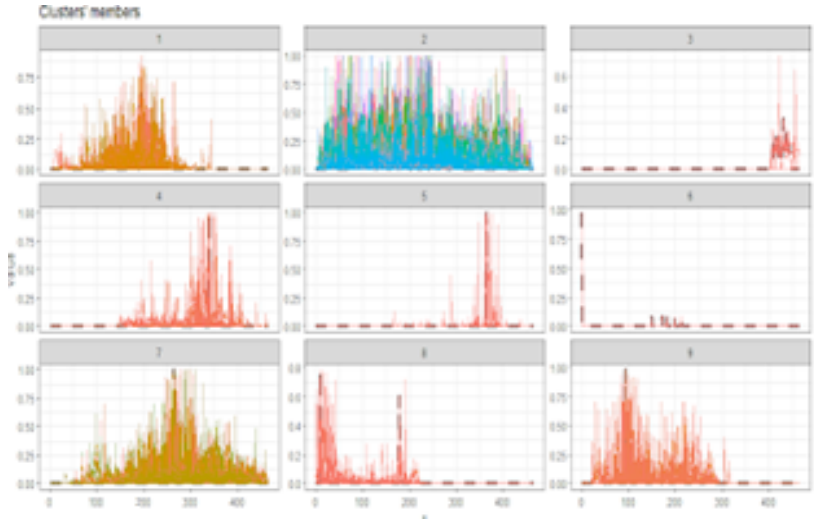


Figure 2.5: Different patterns of #tweet

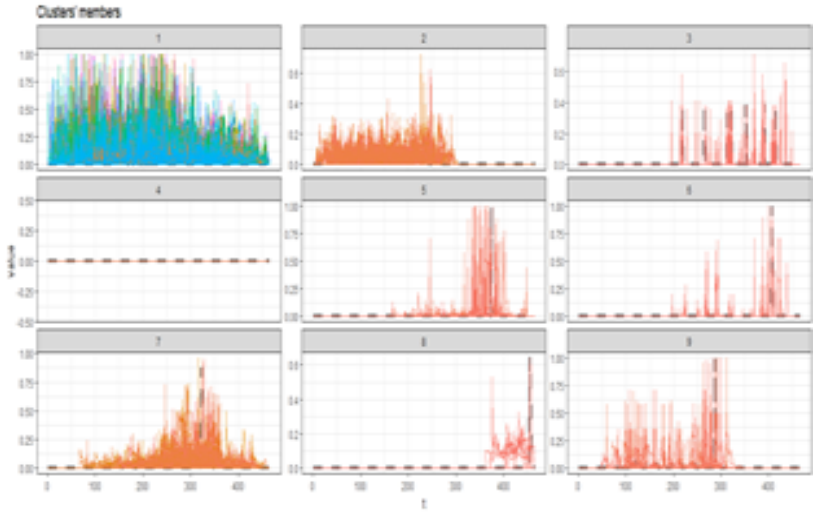


Figure 2.6: Different patterns of #retweet

Table 2.3: CVIs for number of clusters 4 to 10 for negative sentiment percentage

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	0.6030	0.6053	0.6213	0.6125	0.6191	0.6210	0.6074
SF	0.0774	0.0736	0.0316	0.0376	0.0192	0.0165	0.0712
CH	1173.613	880.933	389.815	580.705	498.876	437.810	705.258
DB	1.1849	1.1021	1.0448	1.0928	1.1736	1.1668	1.0562
DBstar	1.4971	1.3908	1.3182	1.4595	1.4987	1.4281	1.3570
D	0	0	0	0	0	0	0
COP	0.3691	0.3686	0.3232	0.3670	0.3246	0.3236	0.3681

values for average positive sentiment score features. Finally, while Table 2.7 shows different CVIs values for neutral sentiment percentage features, Table 2.8 displays different CVIs values for average neutral sentiment score features.

Table 2.4: CVIs for number of clusters 4 to 10 for negative sentiment score

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	0.9134	0.8729	0.8670	0.8638	0.8639	0.8659	0.8754
SF	0.2454	0.0967	0.0673	0.0546	0.0794	0.0708	0.0630
CH	2.49	312.69	250.42	208.90	179.03	156.76	199.36
DB	1.0005	1.9551	1.8339	1.7003	1.5903	1.5151	1.1768
DBstar	1.0919	2.2231	2.1659	2.1380	2.1413	2.0666	1.4107
D	0.2215	0.0381	0.0381	0.03819	0.0381	0.0393	0.0850
COP	0.4760	0.9332	0.9327	0.9322	0.9319	0.9307	0.5248

Table 2.5: CVIs for number of clusters 4 to 10 for positive sentiment percentage

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	5.81	5.83	6.89	5.83	5.80	5.92	5.90
SF	1072.28	0.0561	7343.47	3331.55	3293.53	1152.34	2866.21
CH	986.03	897.18	255.73	571.19	476.55	326.23	286.91
DB	1.8681	1.6435	1.3867	1.6013	1.7256	1.8939	1.4362
DBstar	1.9015	2.2819	1.6278	2.2459	1.9643	2.5989	2.6557
D	0	0	0	0	0	0	0
COP	5.24	4.63	3.51	3.97	3.967	3.52	3.52

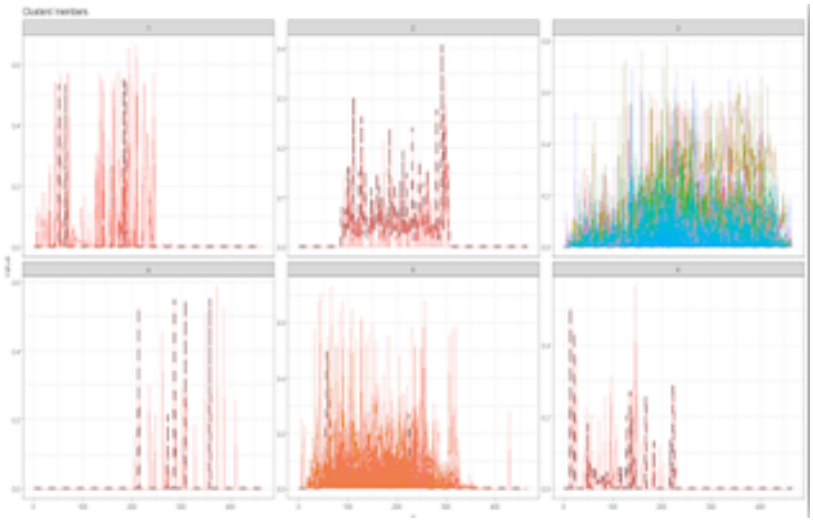


Figure 2.7: Different patterns positive score

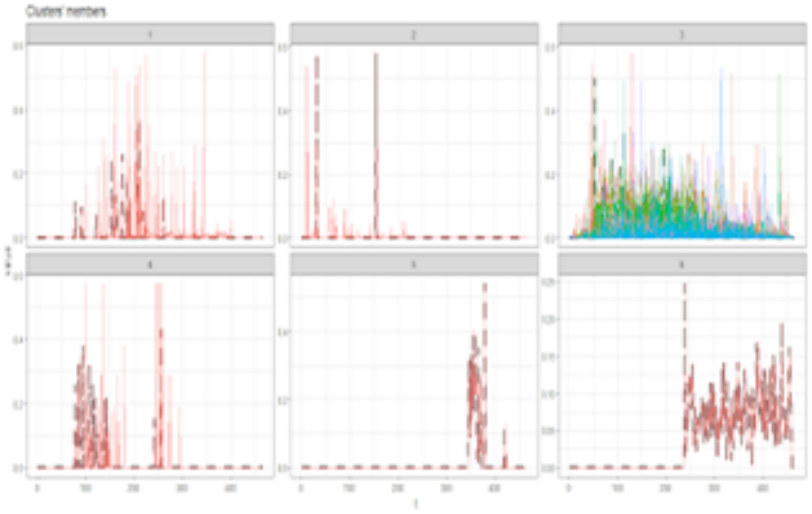


Figure 2.8: Different patterns negative sentiment score

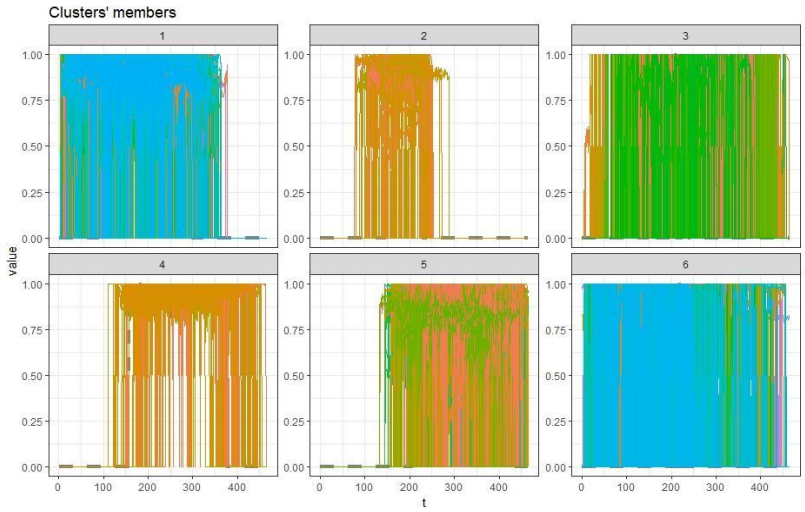


Figure 2.9: Different patterns neutral score

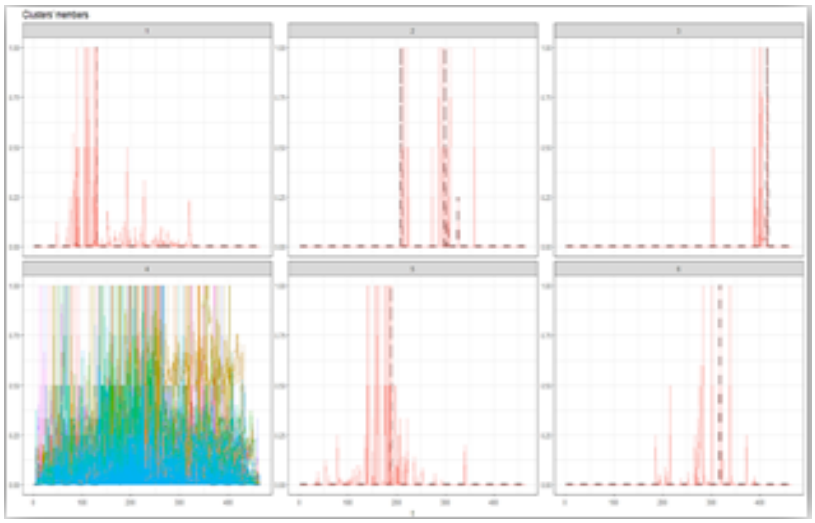


Figure 2.10: Different patterns positive percentage

Table 2.6: CVIs for number of clusters 4 to 10 for positive sentiment score

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	0.02	0.03	0.04	0.02	0.02	0.01	0.01
SF	0.01	0.01	0.02	0.01	0.01	0.01	0.01
CH	67.37	374.56	540.15	234.34	458.59	401.44	357.13
DB	2.6010	1.8376	1.7582	1.9375	2.6280	2.5566	2.4795
DBstar	1.8914	2.0024	1.9870	1.8369	1.8803	1.8183	1.7748
D	0	0	0	0	0	0	0
COP	0.6119	0.6104	0.0607	0.6067	0.6001	0.5796	0.5792

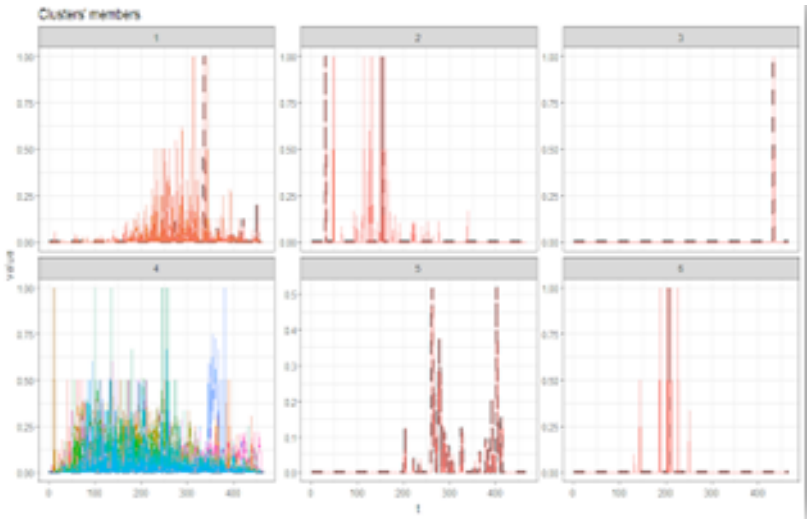


Figure 2.11: Different patterns negative sentiment percentage

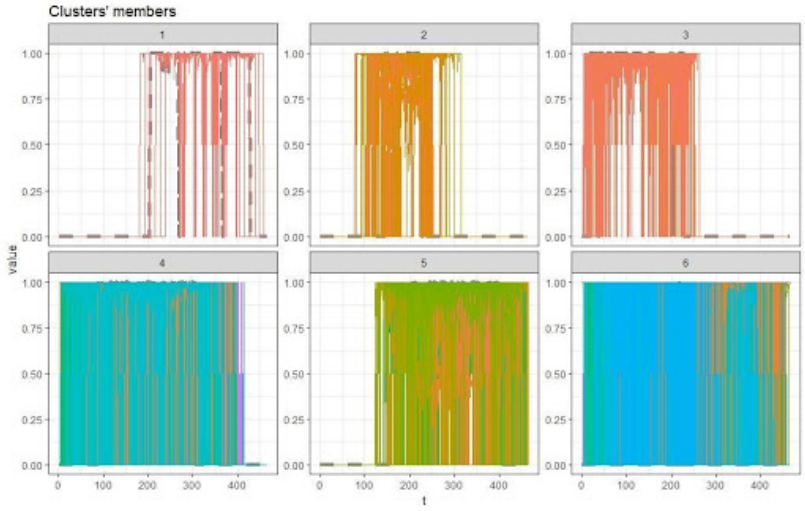


Figure 2.12: Different patterns neutral percentage

Table 2.7: CVIs for number of clusters 4 to 10 for neutral sentiment percentage

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	0.01	0.01	0.02	0.01	0.01	0.01	0.01
SF	$3.78e^{-11}$	$2.16e^{-10}$	$7.41e^{-9}$	$9.66e^{-9}$	$1.30e^{-11}$	$9.05e^{-14}$	$4.06e^{-14}$
CH	$1.40e^{3}$	$1.15e^{3}$	$1.01e^{3}$	$8.57e^{2}$	$7.46e^{2}$	$6.58e^{2}$	$5.41e^{2}$
DB	1.55	1.55	1.39	1.42	3.63	3.98	3.71
DBstar	2.35	2.31	1.86	1.97	4.38	5.14	4.94
D	0	0	0	0	0	0	0
COP	0.44	0.41	0.36	0.37	0.45	0.45	0.44

Table 2.8: CVIs for number of clusters 4 to 10 for neutral sentiment score

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	0.2708	0.1808	0.1666	0.0293	0.0124	0.0391	-0.0544
SF	$1.30e^{-7}$	$8.70e^{-9}$	$5.83e^{-11}$	$1.16e^{-11}$	$3.30e^{-14}$	$1.77e^{-15}$	0
CH	120.85	349.05	752.13	513.69	444.37	389.38	347.64
DB	7.20	7.29	5.11	5.25	7.56	7.74	7.79
DBstar	8.38	8.58	5.55	5.69	9.11	9.57	9.76
D	0	0	0	0	0	0	0
COP	0.40	0.37	0.36	0.39	0.39	0.45	0.44

Table 2.9: CVIs for number of clusters 4 to 10 for direct influenced users

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	0.13	0.11	0.09	0.02	0.01	0.03	0.05
SF	0.02	0.01	0.01	0	0.01	0.01	0
CH	286.13	149.05	52.38	193.69	244.37	139.23	147.64
DB	1.47	2.19	5.21	8.25	23.56	7.24	7.49
DBstar	1.53	4.28	2.15	9.29	3.13	4.52	3.63
D	0.001	0.02	0.05	0.12	0.98	0.34	0.03
COP	0.90	1.47	2.66	1.39	2.69	1.85	2.43

Fig. 2.11, Fig. 2.10, Fig. 2.12 are the visualizations (because there's more than one figure, I assume) of all six different clusters for percentage of positive, negative and neutral sentiment of tweets. In these figures, the x-axis represents the time in hours and the y-axis represents the percentage of sentiments of tweets in each hour. Fig. 2.8, Fig. 2.7, Fig. 2.9

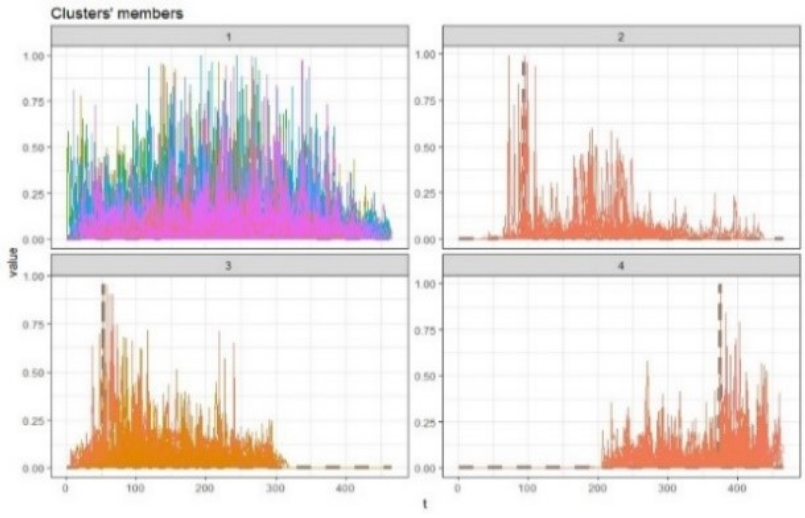


Figure 2.13: Different patterns of direct influence

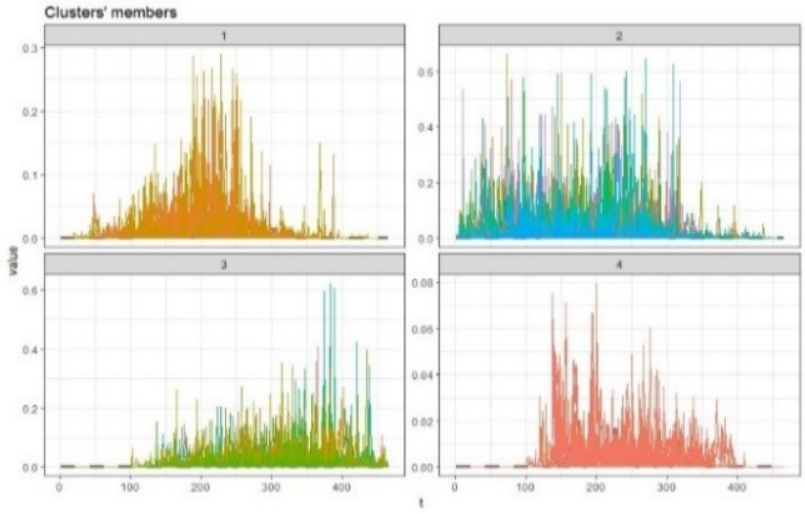


Figure 2.14: Different patterns of indirect influence

Table 2.10: CVIs for number of clusters 4 to 10 for indirect influenced users

CVIs	k_4	k_5	k_6	k_7	k_8	k_9	k_10
Sil	0.07	0.03	0.04	0.03	0.03	0.05	0.02
SF	0.02	0.01	0.01	0	0.01	0.01	0
CH	214.94	79.54	183.28	93.59	124.17	114.32	127.14
DB	1.11	1.02	0.91	0.95	1.06	7.24	7.49
DBstar	1.38	2.18	2.77	1.97	2.23	2.89	1.59
D	$1.1e^{-16}$	$2.1e^{-13}$	$5.1e^{-12}$	$7.1e^{-12}$	$1.1e^{-11}$	$2.4e^{-11}$	$1.3e^{-10}$
COP	0.40	0.91	1.22	1.56	1.45	0.72	2.66

are the visualizations of all six different clusters for positive, negative and neutral sentiment scores of tweets. In these figures, the x-axis is representing the time in hours and the y-axis is representing the sentiment score of tweets in each hour.

Clustering for Network Influence Dimension:

To find out the optimal number of clusters for network influence features, different CVIs of different numbers of clusters from four to ten are compared. Based on the comparison, the best number of clusters for network influence features is four. In Table 2.9 and 2.10, CVIs values for direct and indirect network influence features are displayed. Fig. 2.13 and Fig. 2.14 are the pattern visualizations of all four different clusters for direct and indirect influence features. In these figures, the x-axis represents the time in hours and the y-axis represents the amount of tweets in each hour.

2.4.3 Information Diffusion Patterns Recognition Discussions

The visualization of different patterns from Fig. 2.5 to Fig. 2.14 clearly show that some common patterns are present in many features. Some of the very common and easily explainable patterns are discussed here. In Fig. 2.5 cluster number one is a common pattern. The graph grows slowly over time, reaching a peak before gradually declining; this represents the hashtag with the similar type of popularity growth. A similar type of pattern can be observed in cluster 1 in Fig. 2.14. The second most popular pattern is observed in cluster 4 in Fig. 2.5, cluster 1 in Fig. 2.6, cluster 4 in Fig. 2.11. In all these cases the graphs always show a high value. Regarding the volume of tweets, it can be considered as the group of hashtags that are always popular. Some of the patterns show a very high value in their initial phase and slowly decreases in value over time. In Fig. 2.5 cluster 6, in Fig. 2.6 cluster 6 also shows this type of pattern. In Fig. 2.5 cluster 3 and in Fig. 2.6 cluster 3 exhibits the opposite of the previous pattern. In these cases, the value is low in the initial time and increases with time. Other than these patterns, some of the observed hashtags show spike behavior - the graph suddenly gives a very high value for a very short period of time.

2.4.4 Classification of Information Diffusion Patterns of New Hashtags

The proposed system also supports the method to recognize the information diffusion patterns of a new popular hashtag and to predict its information diffusion features over time. In previous sections, time series clustering processes helps us to identify clusters of patterns for each feature in our information diffusion model. Therefore, these cluster labels can help us to build a classification model to recognize the information diffusion patterns of a new hashtag. In this research, k-NN is proposed to build such a classification model. The following steps will describe the procedure to classify the information diffusion patterns of a new hashtag that need to be analyzed:

- Step 1: Find the DTW distance between the time series data of the new hashtag and all the time series data points in each of the recognized clusters that obtained from 2.4.3.
- Step 2: Find k closest points from each of the clusters.
- Step 3: Find the mean of those selected k points for every cluster.
- Step 4: Find the distance between the new data point and the k mean data points (determined in step 3).

Table 2.11: Comparison of testing RMSE when using ARIMA and LSTM for different Information Diffusion parameters

	ARIMA	LSTM 24 $\times$ 128
#tweet	2.08	0.0089
#retweet	2.08	0.0086
#direct_influence_user	2.04	0.0037
#indirect_influence_user	3.09	0.0038
#positive_percentage	0.89	0.0155
#neutral_percentage	3.48	0.0088
#negative_percentage	0.14	0.0024
#positive_avg_score	0.92	0.01
#neutral_avg_score	3.26	0.0096
#negative_avg_score	1.91	0.0133

- Step 5: Find the closest mean point and assign the new data point (time series data of the new hashtag) to the cluster that has that closest mean point. The assigned clusters are the classification of information diffusion patterns that we need to analyze on new data.

The procedure of building prediction models where ARIMA and LSTM are used is described in the next section.

2.4.5 Predicting Information Diffusion Process by ARIMA and LSTM

As described in the section 2, we employed two well-known techniques, which are ARIMA and LSTM, to forecast the value of the information diffusion time series model. To compare the results between ARIMA and LSTM, we employ the data splitting scheme of 70-30 to divide the dataset. 70% of the total 467 time steps will be used to train the models and then those models will be used to predict the remaining 30%. Root Mean Square Error (RMSE) will be used to evaluate the performance of ARIMA and LSTM models.

ARIMA:

For each subset of information diffusion time series, we use grid search to find the corresponding ARIMA model (p,q,d) for each hashtag. As described above, 70% of the total 467 time steps will be used to estimate the ARIMA models. Then the estimated models will be used to forecast the remaining 30% of time steps. The total RMSE of prediction for each subset of our time series dataset will be the total sum of prediction RMSE of all hashtags.

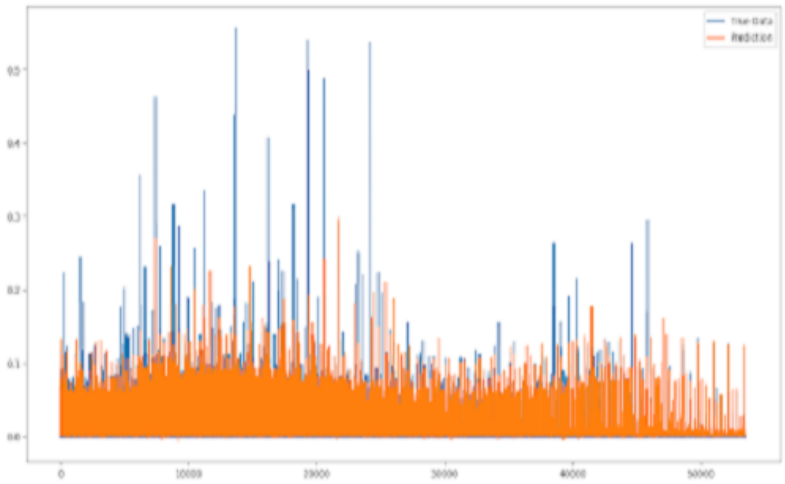


Figure 2.15: The comparison of true value and predictions with LSTM on #tweet dataset

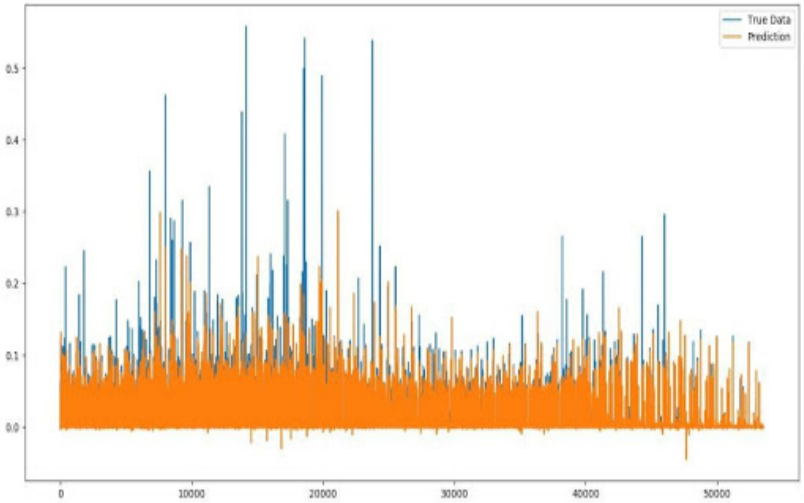


Figure 2.16: The comparison of true value and predictions with LSTM on #retweet dataset

LSTM:

The LSTM model is used for training of each subset of the time series data has two layers. First layer of the LSTM consists of 24 cells and the second layer consists of 128 cells. The model is trained for 100 epochs. The charts from Fig. 2.15 to Fig. 2.19 display the comparison of actual value and the prediction values of LSTM models that correspond with 10 variables in our information diffusion models. Nevertheless, Table 11 displays the performance comparison of different ARIMA and LSTM models that were used to predict our multivariate Twitter information diffusion time series dataset. It can be observed in the Table 2.11 the performance of LSTM model is better than that of AIRMA model while

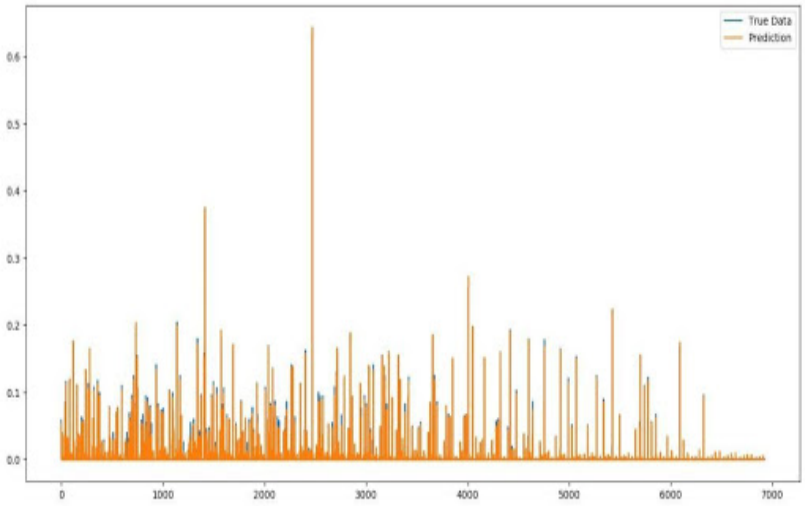


Figure 2.17: The comparison of true value and predictions with LSTM on #negative_sentiment dataset

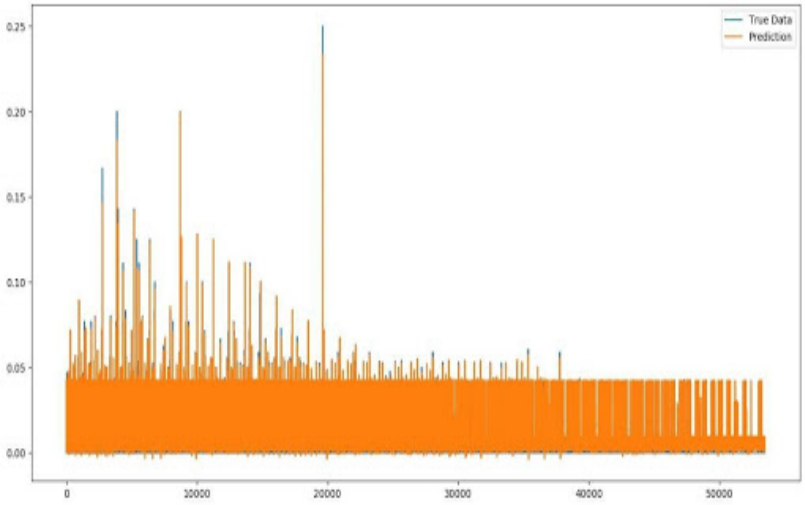


Figure 2.18: The comparison of true value and predictions with LSTM on #neutral_sentiment dataset

predicting the information diffusion patterns. Other than that for each of the pattern LSTM model also performs better than the equivalent ARIMA model.

2.4.6 Improved Technique for Short Text Sentiment Analysis for Information Diffusion on Social Network

In the process of finding better ways to analyze sentiments, we found that tweet replies provide useful information which helps to determine sentiments accurately. Therefore, the difficulty of analyzing sentiments of short texts can largely be overcome by looking

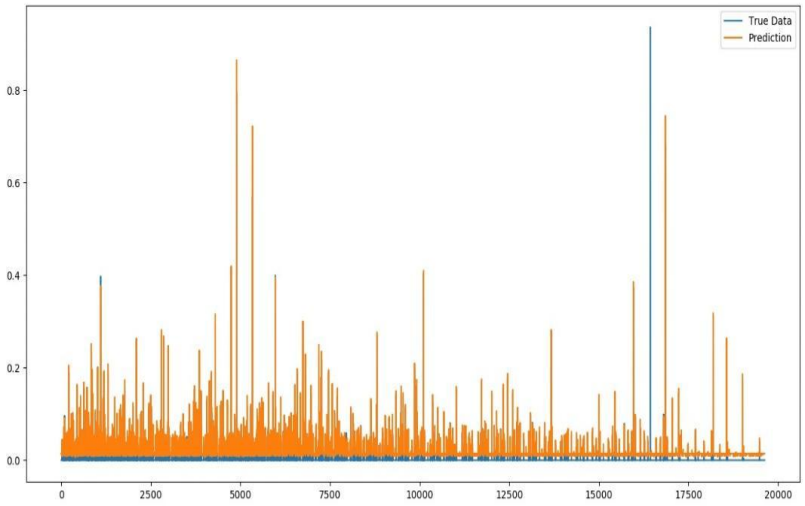


Figure 2.19: The comparison of true value and predictions with LSTM on #positive_negative dataset

into the tweet replies. The tweet is limited to 140 characters, as are replies to tweets, though replies are often less than 50 characters. The sentiment analysis algorithms and libraries are effective for long texts but do not produce good results for short texts, which we have observed in previous research [30]. There are mainly two reasons behind the not-so-impressive performance by the sentiment analysis algorithms and libraries: First, the size of the text is small, and second, the context of replies are not easily known, as the replies are sometimes just one or two words. To deal with these two problems and achieve a better result, the proposed data handling method is designed. We appended replies with their original tweet to serve two purposes. First, as the number of replies increases, the size of text also increases, which facilitates analyzing the sentiment using NLTK or other sentiment analysis methods. Second, the sentiment of each of the replies can be evaluated in the context of its parent tweet. As shown in Table 2.12 for Dataset 2, initially the sentiment of the first tweet is measured manually. This research is published in [111].

Table 2.12: Dataset descriptions

Dataset 1		Dataset 2	
Tweet	Sentiment_1	Tweet	Sentiment_1
Reply_1	Sentiment_2	Tweet+Reply_1	Sentiment_2
Reply_2	Sentiment_3	Tweet+Reply_1+Reply_2	Sentiment_3

The sentiment of the first tweet is Sentiment_1. Then the reply (Reply_1) of tweet is appended with tweet and again the sentiment is measured, which is Sentiment 2. Throguh this method, one by one the rest of the replies are appended with the tweet and in every

occasion, the sentiment is discovered. On the other hand, the Dataset 1 is just individual tweets or replies and their corresponding sentiments. In the rest of the paper, Dataset 1 refers to the dataset where Tweets and their corresponding replies are not appended, and Dataset 2 refers to the dataset where Tweets and their replies are appended.

The corpus collected from Twitter for the entire experiment of information diffusion is huge and comprises 21 gigabytes of data. We selected some required information from the Tweets, TweetId, TextType, TweetText. A set of twelve features for short-text sentiment analysis are extracted from the TweetText of each tweet, described in Table 2.13 below.

Table 2.13: List of extracted features

Features	Description of features	Feature type
Direct message	tweet mentions a specific person	Binary
Includes username	tweet contains usernames	Binary
Includes URL	tweet contains urls	Binary
Exclamation mark	tweet contains exclamation marks	Binary
Question mark	tweet contains question marks	Binary
Term positive	Presence of positive previously defined words	Binary
Term negative	Presence of negative previously defined words	Binary
Emoticon positive	tweet contains positive emoticons	Binary
Emoticon negative	tweet contains negative emoticons	Binary
Positive, Negative sentiments	Positive and negative score using NLTK	Binary
Parts of speech using NLTK	Parts of speech	1 to 35

Extracted Features

SVM and RF algorithms are applied using the twelve extracted features on both the datasets. The entire dataset is divided into training and testing sets with a 7:3 ratio. To implement SVM and RF scikit-learn version 0.19 library [3] is used. For both the algorithms, initially the data is vectorized using the following parameters: vectorizer = TfidfVectorizer(min df=5, max df=0.8, sublinear tf=True, use idf=True, decode error=ignore). In order to implement SVM, linear kernel is used. For RF, 1500 trees are used in the forest. Different numbers of trees are tried ranging from 500 to 3000; it has been observed that the best result is achieved when number of trees are 1500. Performance of SVM and RF using extracted features using both datasets is presented in Table 2.14. Comparing both individual and average accuracy it is clear algorithms using Dataset 2 are performing better than Dataset 1.

Sentiment Analysis using NLTK

NLTK library is used for sentiment analysis for both the datasets. The text data is preprocessed before using the function. The preprocessing is mainly concentrated on substituting a special word or character with a generic term, such as any website (eg. www.abc.com) substituted by "url", any username (eg. abc) substituted by "username", any hashtag (eg. #abc) substituted by "hashtag", all special characters are substituted by "special" and the entire text is converted into lower case. NLTK polarity scores are measured for each input text. Only positive and negative scores are taken into consideration. Table 2.15 represents the result of NLTK, different algorithms with and without the extracted features for both the datasets. Comparing both individual and average accuracy it is clear algorithms using Dataset 2 are performing better than Dataset 1.

Word Embedding with LSTM

Many pre-trained word2vec / Glove models exist, and some of them were trained on huge volumes of data. In this analysis, the one trained with Glove algorithm on over 2 billion of tweets with 200 dimensions is used [119]. In this experiment, we built the sentiment classification model using LSTM with 3 layers: Layer 1, 2, 3 have 128, 64 and 64 cells with 0.25 dropout. The accuracy of the trained model is 0.61. Performance of NLTK and LSTM with Word Embedding model on different topics is presented on Table 2.15.

Table 2.14: Performance of SVM and RF using extracted features using both datasets

Hashtags	Dataset 1		Dataset 2	
	RF	SVM	RF	SVM
Havana	0.964	0.964	0.964	0.964
ImsoOldSchoolThat	0.723	0.766	0.816	0.829
India	0.705	0.746	0.702	0.747
IoT	0.921	0.921	0.813	0.847
iPhoneX	0.087	0.246	0.894	0.881
ITMovie	0.833	0.833	0.901	0.892
LivePD	0.538	0.538	0.752	0.756
Russia	0.62	0.596	0.657	0.664
terrorists	0.576	0.562	0.611	0.67
ThorRagnarok	0.663	0.691	0.687	0.728
trump	0.615	0.598	0.885	0.778
USA	0.736	0.757	0.779	0.778
AVG	0.685	0.665	0.795	0.788

Table 2.15: Performance of NLTK and LSTM using extracted features using both datasets

Hashtags	Dataset 1		Dataset 2	
	RF	SVM	RF	SVM
Havana	0.964	0.964	0.964	0.964
ImsoOldSchoolThat	0.723	0.766	0.816	0.829
India	0.705	0.746	0.702	0.747
IoT	0.921	0.921	0.813	0.847
iPhoneX	0.087	0.246	0.894	0.881
ITMovie	0.833	0.833	0.901	0.892
LivePD	0.538	0.538	0.752	0.756
Russia	0.62	0.596	0.657	0.664
terrorists	0.576	0.562	0.611	0.67
ThorRagnarok	0.663	0.691	0.687	0.728
trump	0.615	0.598	0.885	0.778
USA	0.736	0.757	0.779	0.778
AVG	0.685	0.665	0.795	0.788

Studying the results of Table 2.14 and Table 2.15, it is evident that all the learning machines are producing better results for Dataset 2 than Dataset 1. In other words, the proposed data handling technique is enhancing the performance of sentiment analysis using the popular libraries and algorithms.

2.5 Applications

The increasing usage of social networks provides us a very good opportunity to study social relationships, communities and information diffusion. This work contributes thorough research about analyzing and predicting how information spreads over social networks. Our work can easily be applied in many different real world applications. The two most important are an end-to-end real time analysis application of information diffused over the networks, and an influence on analysis applications such as influence evaluation, influence maximization, etc. In the case of the first type of application real time social network information analysis, based on our proposed methodology, such an application can be built from collecting raw tweets stream data from social networks, preprocessing the raw data, and analyzing the diffusion patterns to predict the diffusion models in the future. Informative real-time visualization graphs can be built, embedded, and displayed on the front-end of such applications to provide up-to-date and meaningful information about the chosen topics (or hashtags). There are two promising application domains: politics and promotion campaigns. In the case the of politics, we can build online web applications to display the social network

impacts of very well-known people such as presidential candidates, senators, celebrities, etc. During political campaigns, such applications can become very popular because they capture popular topics and, more importantly, demonstrate real-time statistics to help experts better understand the current picture of the elections. Nevertheless, in cases of promotional campaign, companies can use such applications to understand the popularity and potential customers' feelings for their products (volume and sentiment in our information diffusion models). Our methodology provides not only real-time analysis capabilities on topics, but also the ability to study and maximize the social influence on such topics. Here, in the second type of mentioned applications, different factors that can affect the patterns and volume of information diffusion can be studied. After that, we can test and obtain the best scheme of factors to maximize the influences. For example, let's imagine that a company wants to increase the popularity of their new product, X. Our models provide such factors can affect the information diffusion model of product X on social networks through elements such as: retweet/replies, number of followers or friends of involved users, influence characteristics of users, etc. Because we already provide the prediction models, we can sketch out different scenarios in which changes are made to different above-mentioned factors and predict the changes in volume of tweets and retweets. Finally, we can select the best scenarios in which the influence is maximized for the target topics (for example product X).

2.6 Conclusion

In recent years, online social media has been increasingly utilized by individuals as well as organizations for a great variety of purposes, including communication, entertainment, marketing, crowd sourcing, political messaging, promotion, propaganda, and fraud. Characterizing, predicting, and quantifying the key aspects of information diffusion processes on social media has, accordingly, become a research topic of growing interest.

The main contribution of our paper is a general approach to recognize the patterns of, and a model to quantitatively predict, information diffusion on Twitter. We first modeled the information diffusion processes on Twitter as a multivariate time series problem in three dimensions (volume, network influence and sentiment) with a total of 10 features. There are two features in volume dimension which are #tweet and #retweet; two features in network influence dimension which are #direct influence users and #indirect influence users; and six features to quantify the percentage and average score of positive-sentiment, neutral-sentiment, and negative-sentiment tweets. We then collected and processed 27.5 million tweets to develop our information diffusion time series dataset with the 10 features. Different temporal patterns of these features were discovered using time series clustering

techniques such as TADPole clustering, hierarchical clustering, and partitional clustering. DTW was used as the distance measure in these clustering techniques.

With the patterns identified, we built an information diffusion prediction model for new topics or memes (hashtags on Twitter). Our prediction model comprises two phrases: the first phrase determines the pattern of hashtags by using k-NN with DTW distance on our clustering result; the second phrase builds the time series forecasting models using a traditional AIRMA approach and the non-linear LSTM approach. We have built different forecasting models with and without using the pattern information. The performance comparison shows that building LSTM models for each cluster resulted in significantly better performance than other models. Therefore, we believe that our method holds great promise to be effective in real-world applications of analyzing and predicting the information diffusion processes of new topics or memes in Twitter. To enhance and refine our proposed model, a better measure of influence (possibly something like influencetracker.com [8]) can be used to draw more inferences than just the count of directly and indirectly influenced people. Second, sentiment analysis methods specifically developed for shot texts or tweets [112], [149] will need to be incorporated to provide accurate measures of the sentiments of tweets. Third, a thorough analysis of network structure and its effect on information diffusion is another important direction for future research. Finally, this model should be applied to other social media platforms [65], [66] to evaluate its performance for validation and future development.

Chapter 3

Social Bots on Twitter

Twitter bots have evolved from easily detectable, unsophisticated looking content spammers and intrusive identities to deceptive key players embedded in deep levels of the social networks, silently promoting affiliate campaigns, marketing premium versions of online products, and orchestrating coordinated political movements. Recently, multiple works on social bots on Twitter have discussed this paradigm shift, moving from building highly accurate machine learning classifiers to identifying individual bots towards focusing on the operations and existence of bots in a collective manner. In this work, we study two different families of Twitter bots which have been studied before for showing spamming activities through advertisement and political campaigns, and perform an evolutionary comparison with the new waves of bots recently identified. We uncover various evolved tendencies of the new wave of social bots under social, communication, and behavioral patterns. Results show that those bots demonstrate evolved core-periphery structure, deeply embedded and robust communication networks, complex information diffusion patterns, heterogeneous content authoring patterns, mobilization of leaders across communication roles, and presence of niche topic communities, which have made them highly deceptive as well as more effective in their operations than their traditional counterparts. Finally, we conclude our work by discussing possible applications of the discovered behavioral and social traits of the evolved bots to build highly robust and effective bot detection systems.

Social network or social media phenomenon began around 2003. In the initial years of social media, MySpcace and later Orkut were the two most popular social media. Later in 2008 Facebook came joined the list and got lots of popularity. Structurally all of them were very similar and their purpose was to get connected with friends and family. In 2006 Twitter came with a different approach, in Twitter people can follow some event, organization or a person based on their choice and get information directly from the source. This opened up a new horizon of information spreading or broadcasting. Apparently, Twitter may seem like a parallel medium of traditional news media but it lacks the authenticity of the traditional news media. The information on Twitter can be truthful, untruthful, or an opinion of the writer. This is one of the downsides of social media, there is no proper way to judge the correctness of the information. Information can be spread by automated programs or bots, social media

also suffers from the problem of fake accounts or impersonators. To identify a real user from fake profile Twitter is providing some verification services to the accounts or user although that is very limited to very popular Twitter users. Moreover, every social media company has its own set of an algorithm to identify any kind of malicious or suspicious activities and stop it immediately. As the algorithms for identifying the bot accounts in Twitter became smarter and sophisticated, the Twitter bots also became very smart. Earlier Twitter bots are used for polluting the content of Twitter feed [88], affiliate marketing [17], and link farming [58] etc. These are repetitive jobs and Twitter account which are involved in this kind of activities are very easily identifiable. Over a period of time, the Twitter bots have changed their types, patterns and range of activities. Now Twitter bots are used for manipulating elections [24], public opinions [55] by spreading malicious content on Twitter. In this article, our discussion is concentrated only on Twitter as Twitter provides different APIs [15] to access the live and historical data.

In 2011, Kyumin Lee et al. published their research work on content polluters on Twitter [14], is one of the earliest research work on Twitter bots and their malicious activities. They observed the content of Twitter for seven months. They observed suspicious Twitter accounts, including an analysis of link payloads, user behavior over time, and followers/following network dynamics. They also evaluate the effects of different features to identify suspicious Twitter accounts or automatic content polluter. This research was a study on different types of Twitter bots whereas Jacob Ratkiewicz et al. [124] research on specifically Twitter bots which are involved in political campaigns. In their research Jacob Ratkiewicz et al. designed a web service which helped to track the political memes on Twitter. This web service also helped to detect astroturfing, smear campaigns, and other misinformation in the context of U.S. political elections. In [27], Yazan Boshmaf et al. discussed four major vulnerabilities which can be a potential cause for a large-scale infiltration campaign. These four vulnerabilities are i) Ineffective CAPTCHAs, ii) Sybil Accounts and Fake Profiles iii) Crawlable Social Graphs and iv) Exploitable Platforms and APIs. This research was performed on Facebook and it gives lots of important insight and propelled this research area to move forward.

There has been recent progress on detection of bots on Twitter, such as the work by Botometer [4] developed by Indiana University Network Science Institute (IUNI) and the Center for Complex Networks and Systems Research (CNetS) which leverages more than one thousand features (based on users, friends, network, temporal, linguistic and sentiment) of tweets and users to classify a Twitter account as a Bot. Similar applications like Botcheck.me [3] and Tweetbotornot [13] are also developed by the University of California, Berkley and Dr. Michael Kearney of the Informatics Institute in the University of Missouri. Botcheck.me

is developed to detect political propaganda and Tweetbotornot is an open-source package. Similar works employ Machine learning methods using features from user profiles [95], graph-based features [148] and temporal features [49], for the classification of malicious and benign accounts on Twitter. However, as the detection algorithms and strict policy to monitor those accounts have improved, the bots on Twitter have also evolved. The spambots have evolved into social bots [40], adopting complex content posting and social interaction patterns. Some of the emerging trends in Twitter bot research have deviated from proposing machine learning features to propose new social dimensions to fight and overcome the new wave of social bots. Some of those new dimensions include the study of lockstep behaviors between user tweets [78], detection of latent group anomalies in graph [160], and similarity between digital DNA sequences [39].

The work of Cresci et al. [40] highlights the paradigm shift of social spambots by introducing a novel dataset of Twitter bots active in three different cases. They extend the idea of analyzing the collective behaviors of social bots, rather than separating individual accounts for malicious behaviors. The new wave of social bots they identified has a very high survival rate with 96.5% of them still being active on Twitter, which demonstrates the highly deceptive capability of the bots. Surprisingly, the new wave of social bots presented by [40] was even able to fool human annotators on crowdsourcing campaigns as the annotators obtained an accuracy of less than 24% in identifying the social spambots with a heavy proportion of False Negatives. Even the state-of-the-art public bot detection service Botometer demonstrated very low recall on detecting such bots. The social bots also remain largely undetected through other techniques of spambot classification like supervised classification, unsupervised classification, and graph clustering-based approaches, which demonstrates the real threat of the new wave of social bots. Despite a relatively dormant spamming behavior of the social bots, those bots were able to generate replies and retweet interactions from genuine human accounts.

3.1 Objective

From the above discussion, it has been observed that recently the Twitter bots have changed their nature. The changes have been observed in their nature of interactions with other Twitter users; their followings on Twitter; type of content and pattern of information diffusion etc. In recent research [40], [78], [160] discussed about these few patterns of Twitter bots. This research is largely inspired by their work and an attempt to find some interesting details about social bots. This research is published in [116]. The major objectives of this study are mentioned below:

A. To contribute to the literature of the new waves of social bots by studying in detail two different types of such bots on Twitter: 1) Political Bots, 2) Advertisement Bots and comparing them with their traditional counterparts for each of the types.

B. To analyze how the social bots have evolved themselves in terms of network structure by answering three important questions: ***RQ1)*** How do the new wave of social bots differ from traditional bots in terms of social network statistics, their organization of Core-Periphery structure ? ***RQ2)*** How embedded are the social bots in their social as well as communication networks? ***RQ3)*** How do the networks of the social bots perform under Robustness attack?

C. To study the information diffusion and communication patterns of the social bots by answering the following questions: ***RQ4)*** How does the information diffusion patterns of the social bots look like? ***RQ5)*** Do the bots have different communication leaders across different forms of communication networks? ***RQ6)*** How homogenous and distributed are the categories of tweets coming from bots, compared to their traditional counterparts?

D. To perform detailed content analysis of the tweets produced by those bots by answering the questions: ***RQ7)*** Do the social bots have any specific patterns of topic distribution over time? ***RQ8)*** Do the bots have some community specific content spreading behavior? Are there any niche topic communities?

E. To discuss possible exploratory network analysis directions and advanced features for machine learning classifiers to detect ever-evolving plethora of novel social bots.

3.2 Datasets Used

Data played a very significant role in this research work. As we are trying to compare the nature of two types of Twitter bots four different datasets have been used. These datasets are containing data for i) Traditional Advertisement Bots, ii) Traditional Political Bots, iii) Social Advertisement Bots, and iv) Social Political Bots. In the rest of this article, the spammer bots are referred to "traditional bots" and the new wave of bots are referred to "social bots." The classification of traditional bots and social bots are done based on the different characteristics discussed earlier in the earlier sections.

3.2.1 Traditional Advertisement Bots

The dataset used by [88] in their research is used for analyzing the behavior of traditional advertisement bots. The Twitter bots are mentioned the dataset are four types of content polluters or spammers, such as: i) Duplicate Spammers, ii) Duplicate Spammers, iii) Malicious Promoters, and iv) Friend Infiltrators. These classification does not segregate the traditional advertisement bots from others. Hence, some careful inspection and data pre-processing have done on this dataset. It has been observed that the content polluters are mostly twitting or re-twitting a group of URLs repetitively for any campaign, propaganda or advertisement. So in the pre-processing step, we identified the Twitter accounts which are spreading the spams only for advertisement purpose. It has been found that a group of Twitter account is spamming URLs related to a particular domain, called "Aweber." "Aweber" is an email marketing service provider. So all Twitter accounts spamming URL for "Aweber." are marked as traditional advertisement bots and used for further analysis of the characteristics.

3.2.2 Traditional Political Bots

The dataset provides by [108] has been used as traditional political bots in the current research. The bots in this dataset tweeted about Arab Spring activity in Libya, from February 3rd, 2011 to February 21st, 2013, and were labelled as bots based upon the deletion and suspension of accounts by Twitter services. All the Twitter accounts mentioned in the dataset are not currently available as most of the users had already been suspended by the Twitter platform. So a subset of the dataset is used in this research.

3.2.3 Social Advertisement Bots

For category of social advertisement bots, social spambots #2 dataset from the work of [40] is used, which consists of manually labeled bots that spent several months promoting a mobile application, called Talnts, using the #TALNTS hashtag.

3.2.4 Social Political Bots

For social political bots, the social spambots #1 dataset from [40] is used. It consists of novel group of social bots being active on the 2014 Mayoral election of Rome employed by one of the runner-ups of the election to publicize his policies.

After expanding the metadata of all the valid tweets and the users, the statistics of the final dataset for the respective categories of Twitter bots under study is shown in Table 5.1.

3.3 Methodology

3.3.1 Tweet Pre-Processing

As mentioned in the last section all the Twitter accounts and the data produced by those accounts are collected from repositories of the previous researchers. This raw data is not suitable for analysis because it contains lots of redundant information. So this raw data has been passed through a few data pre-processing steps. In the first step of the data pre-processing the non-English tweets are translated to English as in the later stage we used few libraries which are applicable only for the English text. The majority of the non-English tweets were in Italian language and Cloud Translation API from Google Cloud [6] is used to convert them into the English language. As tweets are normally written informally so lots of tweets contain emoticons, special characters. Hence, in the second step of the data pre-processing all emoticons, special characters, low-frequency words, stop words, HTML tags, and URLs are removed. In the last step of data pre-processing all characters are converted to lower case for uniformity of dataset, then lemmatized them and expanded common English contraction words to clean the text for Topic Modeling in the next step and Topic Over Time analysis for the later stage of this study.

3.3.2 Topic Models

Once the tweet data is cleaned, Python's Gensim library [126] has been used to build LDA topic models. This model is built for 10 topics and 100 iterations are used to build it. The value of the parameter α (alpha) is adjusted by the LDA algorithm of the Gensim package. In the next step all the pre-processed tweets are classify to these 10 topics.

3.3.3 Creation of Social Interaction and Communication Networks

After data cleaning and topic modeling, five social interaction and communication networks are created using the Tweeter users details are obtained from the datasets. These networks are 1) Social Network, 2) Retweet (RT) Network, 3) Mention Network, 4) URL Network, and 5) Hashtag (HT) Network.

A. **Social Network:** is a directed network where there is an edge from a Bot A to Bot B if Bot A follows Bot B.

B. **RT Network:** is an undirected network where there is an edge from Bot A to Bot B if both of them have retweeted *thres_rt* or a higher number of similar Twitter users.

C. **Mention Network:** is an undirected network where there is an edge from Bot A to Bot B if both of them have mentioned *thres_mention* or a greater number of similar Twitter users.

D. **URL Network:** is an undirected network where there is an edge from Bot A to Bot B if both of them have tweeted *thres_url* or a higher number of similar unique URLs.

E. **HT Network:** is an undirected network where there is an edge from Bot A to Bot B if both of them have tweeted *thres_ht* or a higher number of similar unique Hashtags.

Once the networks or the graphical representation of the networks are created it has been observed that the edges of the graphs do not have any weight. It suggests that any two nodes in a network have the same degree of correlation. This assumption is not valid and may cause many wrong results in the later part of the experiment. Moreover, the networks are very dense which is sometimes not easy to manage. To solve these two problems, slicing [169] is applied to these networks to eliminate the weak ties or low strength edges. In practice, the user selects a cut-off threshold of T and the value of T determines the density of the resulting graph. The weights of the edges are then compared with the threshold and it the weight of the edge is less than the threshold then the edge is eliminated. This way the graphs can be sparse and manageable. Before applying the slicing method, the edges should have weights. To solve both the problems NetworkX [63] library of python is used. Using this library the weight calculation of each edge can be done and edges with less weight than the threshold can be eliminated. To use NetworkX, value for T needs to decide at first and then routine provided by the library can be implemented. The best value of T is determined by the trial-and-error method. If the value of T is too low, then we may not find any significant difference from the initial network. If the value of T is too high then the resultant network may appear as a collection of small fragmented networks which loses lots of significant information. The trial-and-error process is followed for this research is mentioned below:

A. Select a T

B. Apply slicing method on the network

C. Got some measurements as depending on the requirement of the experiment.

D. If the measurements are not satisfactory then go back to step 1.

For the relatively sparse networks of traditional advertisement bots, social advertisement bots and traditional political bots, we used the threshold of 5 for *thres_rt*, *thres_mention*,

thres_url and *thres_ht*. For a relatively denser network of social political bots, we used thresholds of 20, 30, 20 and 50 for *thres_rt*, *thres_mention*, *thres_url* and *thres_ht* respectively. Throughout the rest of the paper, RT Network, Mention Network, URL Network and HT Network are collectively referred as communication network of the bots.

3.3.4 Complex Network Analysis

We used the NetworkX library for most of our algorithmic computations related to Complex Network Analysis such as Core-Periphery Analysis, K-Core Decomposition, Centrality Leaders Correlation. For Robustness Analysis of the social and communication networks, we used the the open source implementation by [76] and adopted the sequential targeted attack approach. In this approach, centrality measures (Degree, Betweenness, Closeness and Eigenvector) is calculated for all the vertices in the initial network, and the vertex with highest centrality measure is removed. The centrality measures are recalculated for all vertices in the new network and the highest ranked vertex is removed, with the process repeating until desired fraction of vertices has been removed. For analyzing Information Diffusion and Content Authoring patterns, we extract required user level and tweet level metadata of the retweet timeline from the official Twitter API.

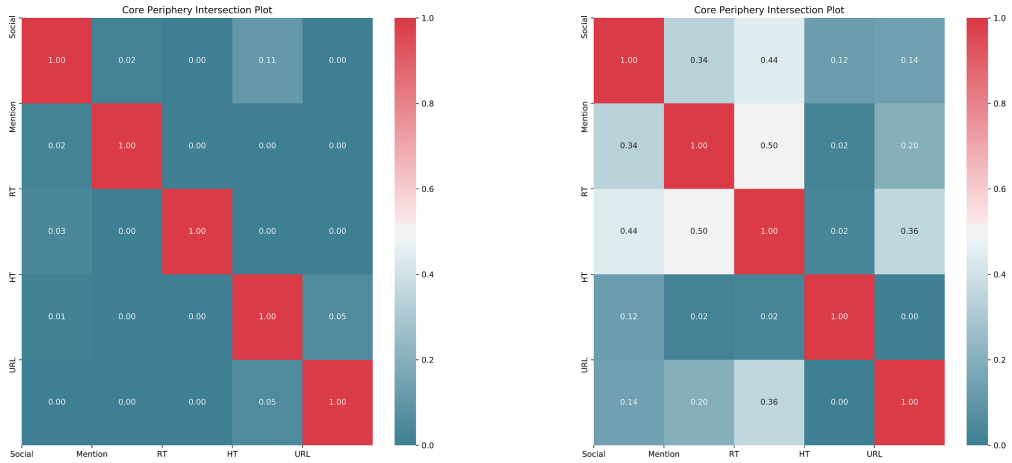
3.4 Results

3.4.1 RQ1: Basic Network Statistics and Core Periphery Structure

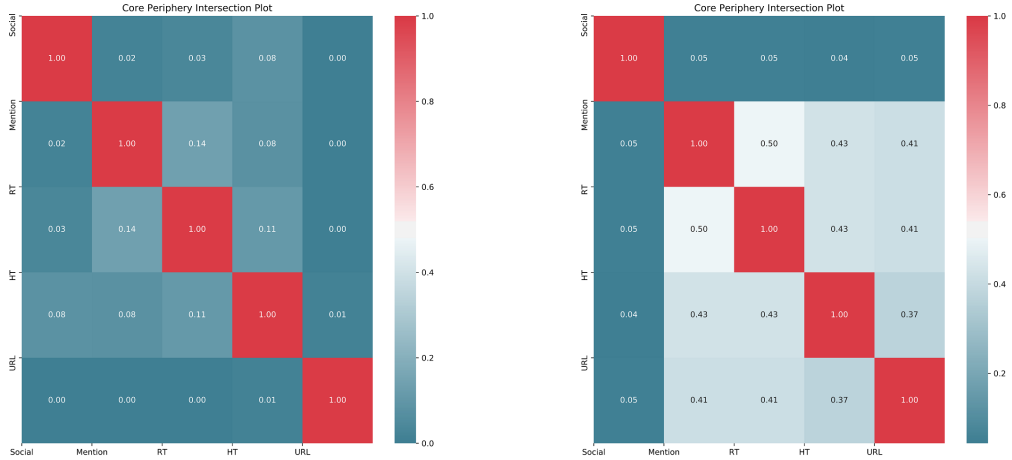
For advertisement bots, the social network of the traditional bots is more densely connected, has higher clustering and transitivity than that of social bots, while the other communication networks are very sparse, with lesser clustering and transitivity values. In the case of social bots, the other communication networks (RT, Mention, URL and HT) are denser, have higher clustering and transitivity than their traditional counterparts. In the case of Political bots, the social bots are denser, highly clustered and more transitive than their traditional counterparts in their organization of communication networks, as well as the social network.

The social and communication networks of both types of traditional bots have a smaller core and a larger periphery which is connected very weakly to the core. The social bots in both cases have a larger, strongly connected network core and a relatively smaller size of peripheral nodes which are strongly connected to the core. The results demonstrated a close-knit and more focused network structure in the social bots, closer to the findings of human communication networks discussed in the work of [82].

Next, we wanted to study if the core nodes operating on the network structure remain stable across the communication networks by plotting the intersection of the members of



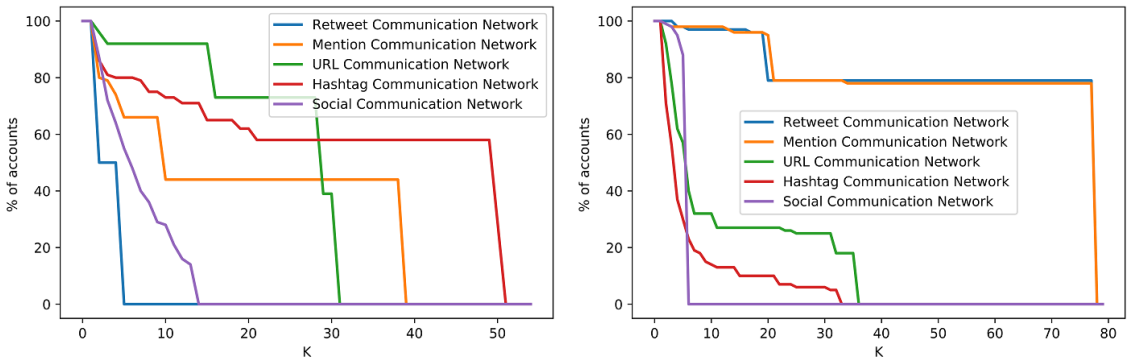
(a) i & ii



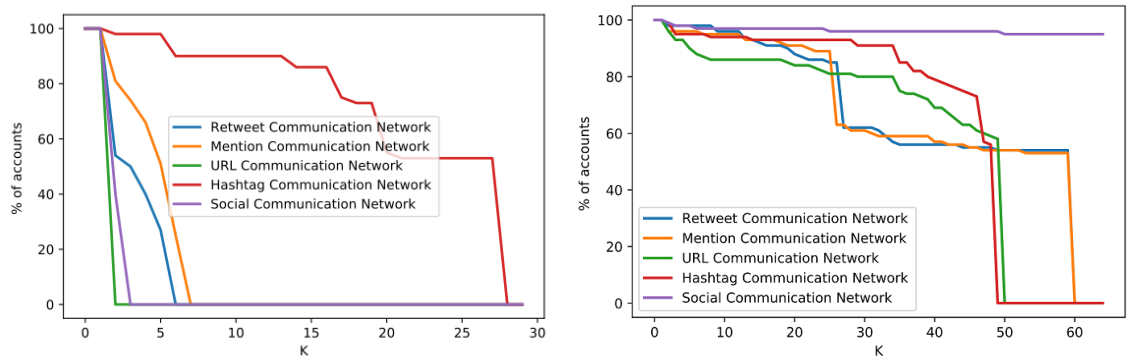
(b) iii & iv

Figure 3.1: Network Core Members Intersection Plot of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots

core on each of the network. As seen in the Core-Periphery intersection map in Figure 3.1, we found that the core nodes in the social bots remain more intact across the different communication networks, whereas there is very little intersection between the core nodes of communication network, in case of traditional social bots, both in the case of advertisement bots and political bots. This demonstrates the evolution of social bots towards stable sets of principal actors in the central core structure, across the different communication channels.



(a) i & ii



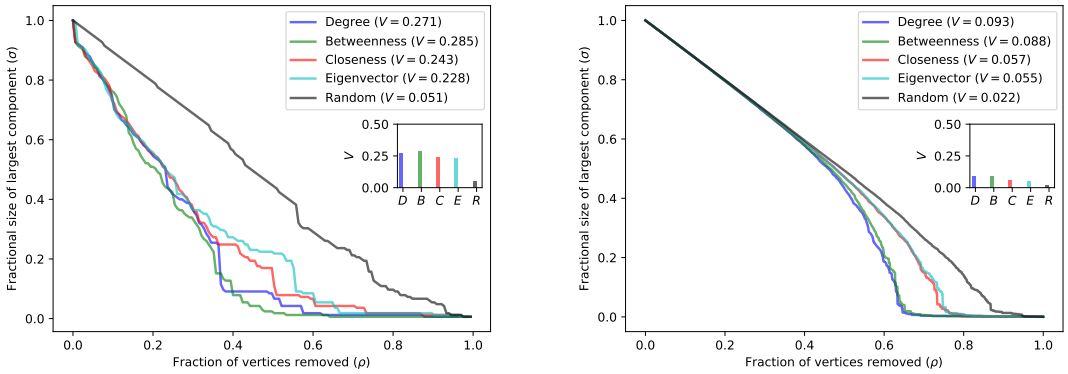
(b) iii & iv

Figure 3.2: K-Core Decomposition Analysis of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots

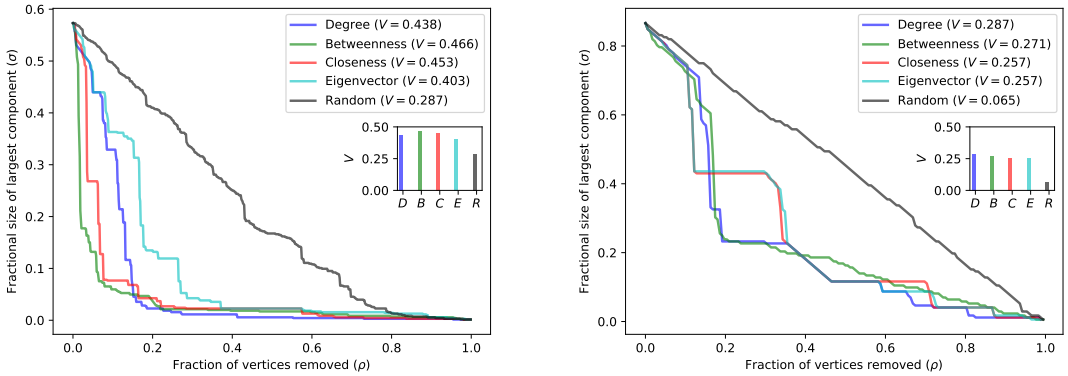
3.4.2 RQ2: K-Core Decomposition

We applied a commonly used method in graph theory, K-Core decomposition to study the structure and embedding of nodes in the graph. In Figure 3.2, we plot the percentage of accounts retained (y-axis) as we increase the value of k in the K-core (x-axis).

When we compare the social networks of traditional advertisement bots and social advertisement bots, even after considerable amount of increased core size of the traditional bots, much of the network core is retained, signifying the deeply embedded bots in their social networks, whereas in a contrasting manner, the social graphs of social bots appear to disrupt easily with slight increase of k. When we observe the decomposition graphs of communication channels, social bots outnumber traditional bots, in terms of proportion of retained accounts. The results are similar with the case of political bots demonstrating higher values of graph degeneracy for the communication networks. This behavior is an evolved tendency of social bots to populate areas of the communication networks being



(a) i & ii



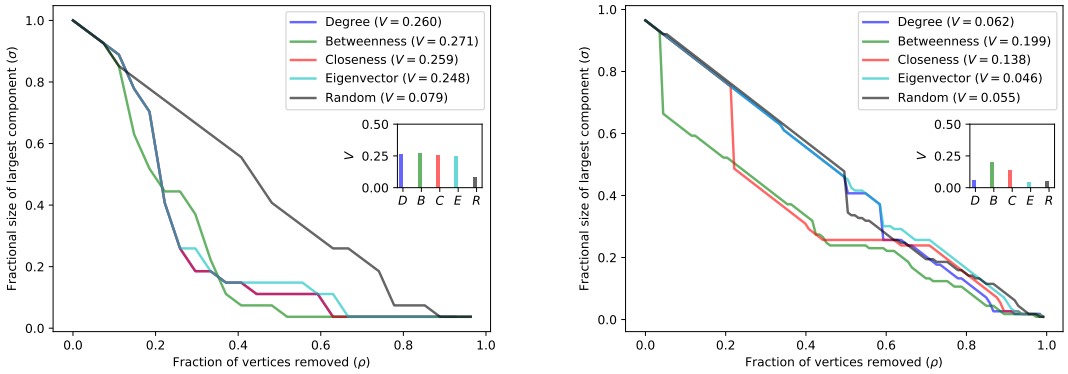
(b) iii & iv

Figure 3.3: Robustness Attack Test i) Friendship Network of Traditional Advertisement Bots ii) Friendship Network of Social Advertisement Bots iii) Hashtag Network of Traditional Advertisement Bots iv) HashTag Network of Social Advertisement Bots

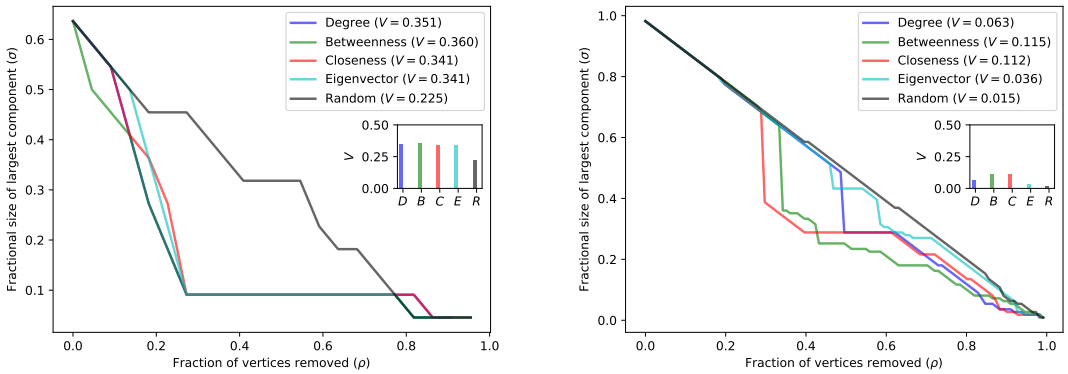
more central and better connected whereas being shallowly embedded in their social graphs. From the bot design economy point of view, it is easy for bots to get deeply embedded in their social networks, while equally costly in terms of large scale botnet detection. Whereas, disrupting the bots through multiple dimensions of communication channels is more costly and requires more effort from a platform moderator or a bot detection service.

3.4.3 RQ3: Network Robustness

We subjected the five different networks for all of the bots, traditional as well as social, used in our study to Robustness Analysis. In Figure 3.3 and Figure 3.4, we plot the fractional size of the largest component of the network against the proportion of removed network nodes in decreasing order of various centrality measures. The results show that social advertisement



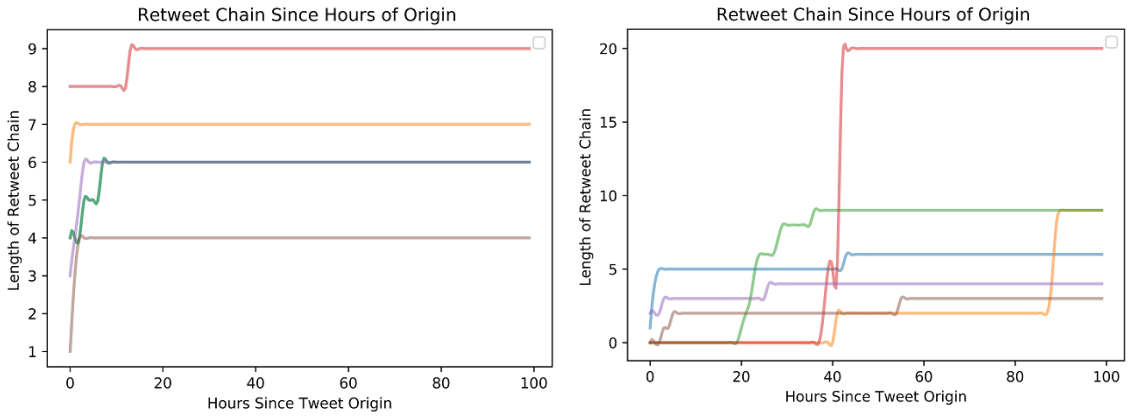
(a) i & ii



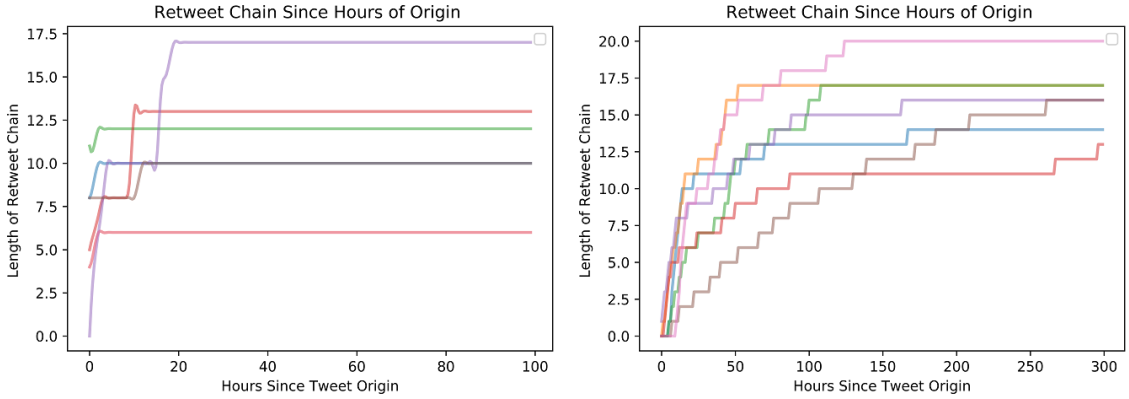
(b) iii & iv

Figure 3.4: Robustness Attack Test i) Mention Network of Traditional Advertisement Bots ii) Mention Network of Social Advertisement Bots iii) RT Network of Traditional Advertisement Bots iv) RT Network of Social Advertisement Bots

bots are very resilient to network robustness, with significant proportion of size of largest component dropping only after more than 60% of its vertices being removed. We can also see that there is no specific centrality vulnerability in the social graph of them as removing nodes based on different types of centrality leaders result in similar effect on the decrease on size of largest component. Whereas, the size of the largest connected component in the social graph of traditional bots decreases substantially, before even 40% of the vertices are removed. The social graph of traditional advertisement bots is very prone to network failure, especially when the centrality leaders on Closeness and Eigenvector are attacked first. The other communication networks are equally vulnerable to robustness attacks through identical centrality leaders. Similarly ,the outcomes of robustness attacks are identical, for traditional political bots and social political bots, with all forms of networks of social bots showing



(a) i & ii



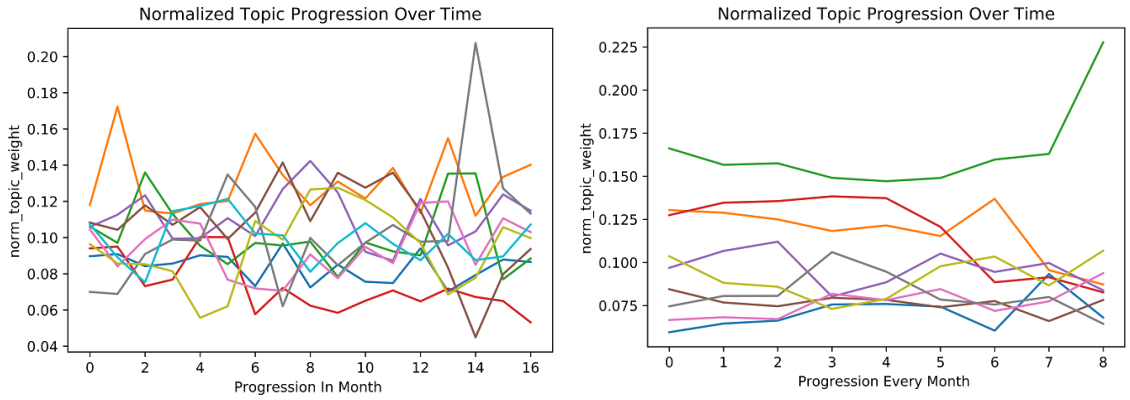
(b) iii & iv

Figure 3.5: Retweet Diffusion Timeline of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots

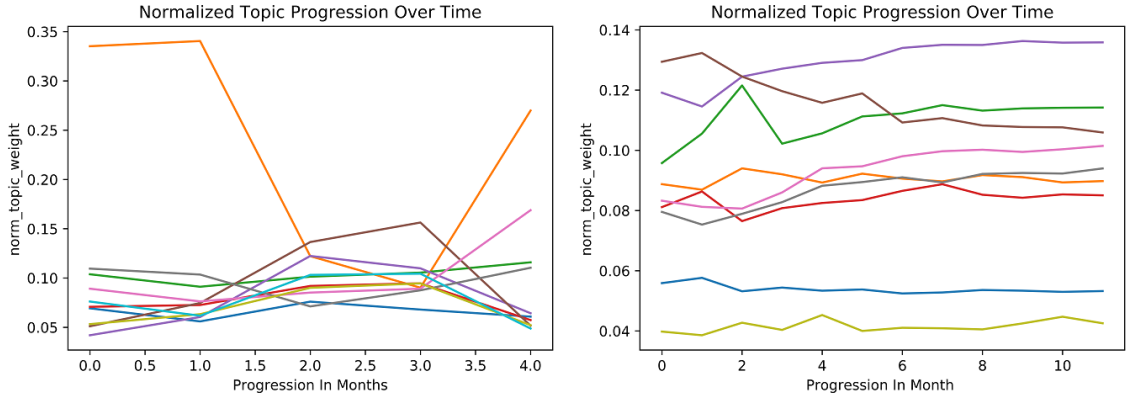
more resiliency to robustness attack than their traditional counterparts. An observation we noted is that amongst the communication networks of social political bots, the Mention network are relatively less robust, through the point of attack of Betweenness Centrality, while the Hashtags network of social advertisement bots are relatively more vulnerable to network failure .

3.4.4 RQ4: Information Diffusion

We collected the top 10 retweeted tweets from each of the bot datasets in our study, which has at least a single occurrence of the campaign related hashtags, or URLs and which were retweeted more than 10 times by user accounts outside of the bot datasets. We then expanded the information diffusion timeline of the tweets by collecting information of the metadata of their retweets. We made sure our study analyzes the tweets that are primarily authored by



(a) i & ii

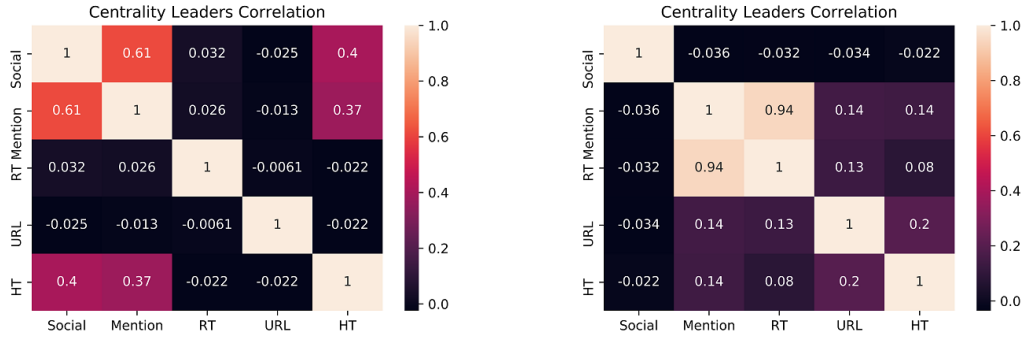


(b) iii & iv

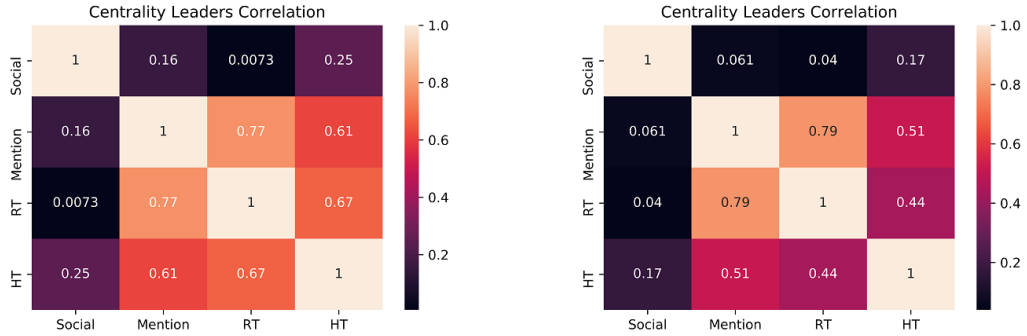
Figure 3.6: Normalized Topic Over Time Distribution of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots

the bots under study, by inspecting the "retweet_status_id" property of the tweet and expand the diffusion network. We visualize the graphs of the information diffusion of different botnets under our study by plotting the cumulative frequency of retweets originated by a tweet against the time (in hours) since the origin of the tweet in the x axis. The steeper the line, the faster the information was spread, and the height of the line (y axis) indicates the number of times it was retweeted. The graphs for the respective botnets are in Figure 3.5

We compared the information diffusion graph of each of the category of social bots with their traditional counterparts. The lines on the diffusion graph of traditional advertisement bots shows fast and abrupt diffusion of information shared by very few actors. The tweets gathered a certain number of retweets within the first hour of their origin abruptly but were unable to gather further retweet along with time. Contrastingly, the social advertisement bots were very effective in diffusing information, showing both the patterns of slow diffusion



(a) i & ii



(b) iii & iv

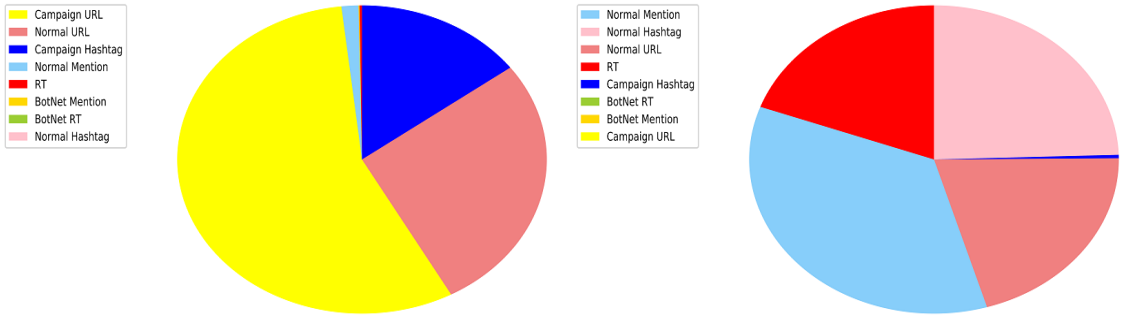
Figure 3.7: Centrality Leaders Correlation Plot of i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots

occurring over a long period of time, shared by a few actors as well as fast diffusion of information shared by many actors. The results were identical in the case of traditional political bots, as the political bots were able to generate a even greater number of multi-user, fast and sustained retweet chains of information diffusion of the social political bots.

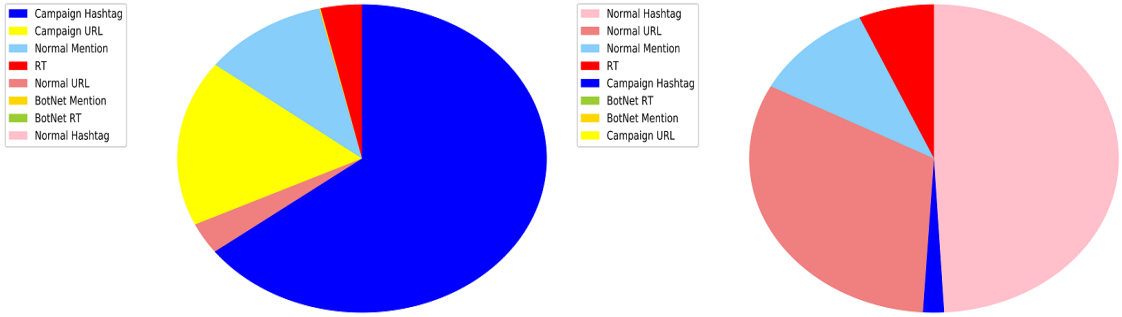
3.4.5 RQ5: Topic Usage Over Time

In this study, we first built the topic model over the pre-processed tweets and divide all the tweets from the respective datasets into equal monthly time buckets and inferred the tweets of those buckets against the learnt topic models. The normalized topic distribution weights of the tweet buckets (y-axis) with the increasing time (in months) since the initial tweet of the respective bot dataset is displayed in Figure 3.6

We compared the social bots with their traditional counterparts on the evolution of topics over time. As shown in Figure 3.6, the normalized topic distribution weight for almost all



(a) i & ii



(b) iii & iv

Figure 3.8: Content Authoring Homogeneity Chart of: i) Traditional Advertisement Bots ii) Social Advertisement Bots iii) Traditional Political Bot iv) Social Political Bots

topics of the traditional advertisement bots increases and decreases abruptly over time. The transition between time buckets of the topic distribution not being smooth, with lots of edgy crests and falls, reflects on the bot's instability on their topic's distribution throughout the time. Contrastingly, the topic distributions for the advertisement networks are distributed evenly across time. The results on this study are similar in the comparison of social political bots and their traditional counterparts, demonstrating the topical stability of the social bots over time.

3.4.6 RQ6: Leaders Across Communication Networks

We examined if the leaders of centrality in the communication and social networks are consistent across the different communication networks, or new communication leaders arise within the communication networks. We calculated the Betweenness Centrality of the communication networks for each type of bots and calculated Kendall's correlation coefficient between the centralities of bots in the networks. Kendall's correlation plot allows us to visualize the ranked correlation between the leaders of the communication networks.

We expected the modern wave of social bots to have negative or very little correlation of rank centrality across the communication networks. Our results show that in the case of traditional advertisement bots, they have positive values for correlation between social communication network leaders and leaders of other forms of communication. Whereas, other forms of communication networks show increased positive correlation between each other than the social advertisement bots. The results are similar in the case of traditional political bots and social political bots, as shown in Figure 3.7. This lack of correlation between signal centralities leaders shows that the new wave of social Twitter bots deploy different role leaders across different communication channels to meet their campaign objectives, while the same leaders of social networks reoccur in the different communication networks in case of traditional bots.

3.4.7 RQ7: Content Authoring Homogeneity

We analyze the tweeting behaviour of traditional bots and social bots to observe how they differ in their strategy of mixing between campaign-related tweets, sharing of spam URLs and normal user tweets to effectively avoid detection. For this study, we divided all of the tweets of the bots into different categories, as 1) Botnet Retweet: retweets of tweet from fellow bot of the campaign 2) Normal Retweet: retweets of tweet from bots outside of botnet 3) Botnet Mention: tweets mentioning fellow bots of the campaign 4) Normal Mention : tweets mentioning users outside of botnet 5) Campaign URL: tweets containing URL related to campaign 6) Normal URL: tweets containing URLs not related to the campaign. 7) Campaign Hashtag: tweets containing hashtags related to campaign 8) Normal Hashtag: tweets containing non-campaign hashtags.

As seen in Figure 3.8, most of the tweets from traditional bots are related to spreading campaign-related URLs and campaign-related hashtags. They very rarely tweet and forward normal tweets, and rarely interact with normal users. Whereas, the social bots, which are deployed for a similar mission to spread advertisement of a product, or propaganda content, have a much-distributed content mixing patterns. The proportion of campaign-related tweets, both Hashtags and URLs in their timeline is very less, as compared to the traditional bots. We can conclude that this very strategy allowed those bots to be more effective in sustaining long time on the community as well as effectively spreading their campaigns at the same time attracting genuine human interactions.

3.4.8 RQ8: Niche Topic Community

Finally, we studied if the bots are spread–out across communities who intermingle their campaign tweets with tweets related to niche topics. When bots tweet not only their campaign related tweets, but also about specific topics, they are targeting a portion of community audience related to the topics, while occasionally disseminating the tweets related to their campaign, allowing them deceptive as well as influence capabilities at the same time. In case of political bots, we also study if there are topical communities of twitter bots to focus on spreading different aspects of the campaigns. We applied Louvain [43] community detection algorithm on the networks of the bots under comparison and separated them into communities. We then extracted the hashtags used by the users of the top communities of the networks and investigated the top hashtags from each community.

Table 3.1: Statistics about the datasets of categorical bots

Category	Tweets	Bots
Social Amazon Spammers	428543	3458
Traditional Advertisers	37922	165
Social Advertisers	1418558	465
Traditional Political Spammers	1967	152
Social Political Spammers	1610016	992

Table 3.2: Traditional Advertisement Bots

Comm 1	Comm 2	Comm 3	Comm 4
newsletter	emailmarketing	EmailMarketing	emailmarketing
photography	integrations	Autoresponder	integrations
LearnPhoto...	EmailMarketing	FreeEbook	marketing
eBook	marketingtips	SocialMedia	email
#1	cmworld	FreeTrial	networkmarketing

Top Hashtags used by the top communities of the advertisement bots in each of the communities are almost similar (newsletter, Email Marketing, Marketing Tips). We can observe one specific community, tweeting numerals (#1, #2, #3) as their top tweets. Upon investigating those tweets, we found that those tweets were again related to Email Marketing tips (Tip #1, Tip #2). The top hashtags across the communities are fairly suspicious with regards to the intent of the deployed bots. Whereas, the top hashtags adopted by the social

Table 3.3: Social Advertisement Bots

Comm 1	Comm 2	Comm 3	Comm 4
TALNTS	ReekSpeaks	TALNTS	VoteUKArianators
WeLoveJustin	TALNTS	EDM	BaNvAfG
music	gossipgirl	EDMSoundofLA	WorldCuP
HBDJustinBieber	Halloween	iPromoteYou	MARCH
myxmusicawards	TheOriginals	hiphopmusic	DubaiWorldCup
nowplaying	ISM2014	hotnewhiphop	LincolnTrialsAtWolves

bots are different across different communities. We can observe a community dedicated on tweeting about a particular music artist (Community 1), another community which tweets about the public event happening at that time (Community 4), and also a community tweeting about web series (Community 2). One interesting observation we found was a specific community (Community 3), which interacted with a very dedicated community of Indie Music artists. As reflected on their hashtags, they tweeted about promoting Indie Rap Music, and EDM sound, with the effort to recruit premium members inside their application, #TALNTS. Compared to their traditional counterparts. We also noted that their primary hashtag for promotion of their app, #Talnts, is ranked lower and dominated by some other hashtags in two different communities. Similarly, the top hashtags across the community of traditional political bots are similar across the different communities, and they are solely related to the campaign they are functioning for, including hashtags which are potentially sensitive to the community.

Whereas, in the case of social political bots, the top hashtags across the communities, as well as the position of the top campaign related hashtags vary. Some communities have some external events as their top hashtags. Alongside mixing of campaign related hashtags with external hashtags, upon cross referencing the top hashtags with the event context, we found that the political bots have some communities dedicated for special sub campaigns within the larger political campaign. For example, Community 1 tweeted mostly about leaders and political figures associated with the event, Community 2 tweeted about senate related activity, Community 3 tweeted about an external event, related to other political turmoil in Palestine, mixed with their tweets whereas Community 4 was focused more on tweeting slogans and political ideologies.

Our observation of community-based Hashtag study demonstrates how the social advertisement as well as political bots effectively deploy niche topic communities. Alongside tweeting about their primary campaign hashtags, in an attempt to remain undetected as well

as gain a level of mutual trust in the community they belong to, they tweet about specific topics and interact with specific communities. We were also able to discover sub campaign related communities within the bots, which intermingle their sub campaign objectives alongside varied, genuine looking tweets.

3.5 Towards new dimensions for Exploratory Social Bots Detection

The comparative differences of the new wave of social bots with their traditional counterparts helped us to gain more understanding on the strategy they adopt to remain unnoticed for a long time, the communication channels they utilize to gain interaction from normal human users. Moreover, the understanding we gained by the evolutionary traits of these bots can be used on future to detect more robust tools to detect similar waves of social bots, which might have remained unnoticed till now. From the viewpoint of an exploratory detection of social bots, we demonstrated that graph-based studies like K-Core decomposition of various signal-based networks can uncover the bots through communication mediums utilized by them. Similarly, the depth of Information Diffusion trees, augmented by the temporal mining of retweet patterns could also be an equally effective further avenue for research.

The study of network robustness attacks on various communication networks of the groups of social bots suggests to us that certain signal networks, like Mention and Hashtag (HT) network could be attacked from vantage points of centrality leaders, to test the resilience of the network, and possibly explore the channels of communication that could be blocked to minimize the effects of the social bots. Based upon the economy of social bot design and objectives, we can also argue that the intersection of Core-Periphery structure across the communication networks can be another pattern to study for the presence of bots. From the network centrality point of view, relative ranked positioning of the centrality leaders across the social network as well as the communication networks could help us in answering the Information Diffusion setup, even for the more sophisticated of the bots. Expanding the graph-based analysis, we also identified behavioral traits of the social bots, from a content analysis point of view. We studied how the new wave of social bots demonstrate human like content patterns on their tweets, by tweeting with similar intensity about a similar distribution of topics for a long amount of time. The homogeneity of the tweets emitted from the bot and the presence of niche topic communities for the promotion of campaigns should be studied in depth for the bots who are continuously evolving and fighting against the adversaries of bot detection systems.

3.6 Conclusion

We respond to the recent call for exploring new dimensions to study social bots by analyzing the behavior of novels social bots under different graph based, behavior based, content based and interaction-based studies. The comparative differences of the new wave of social bots with their traditional counterparts helped us to understand the evolutionary traits of the social bots and gain more understanding on the strategy they adopt to remain unnoticed for a long time, the communication channels they adopt to gain interaction from normal human users. Moreover, the understanding we gained by the evolutionary traits of these bots can be used on future to detect more robust tools to detect similar waves of social bots, which might have remained unnoticed till now. From the viewpoint of an exploratory detection of social bots, we demonstrated that graph-based studies like K-Core Decomposition of various signal-based networks can uncover the bots through communication mediums utilized by them. Similarly, the depth of Information Diffusion trees, augmented by the temporal mining of retweet patterns could also be an equally effective further avenue for research.

The study of network robustness attacks on various communication networks of the groups of social bots suggests to us that certain signal networks, like Mention and Hashtag network could be attacked from vantage points of centrality leaders, to test the resilience of the network, and possibly explore the channels of communication that could be blocked to minimize the effects of the social bots. Based upon the economy of social bot design and objectives, we also learnt that the intersection of Core-Periphery structure across the communication networks can be another pattern to study for the presence of bots. From the network centrality point of view, relative ranked positioning of the centrality leaders across the social network as well as the communication networks could help us in answering the Information Diffusion setup, even for the more sophisticated of the bots. Expanding the graph-based analysis, we also identified behavioral traits of the social bots, from a content analysis point of view. We studied how the new wave of social bots demonstrate human like content patterns on their tweets, by tweeting with similar intensity about a similar distribution of topics for a long amount of time. The homogeneity of the tweets emitted from the bot and the presence of Niche topic communities for the promotion of campaigns should be studied in depth for the bots who are continuously evolving and fighting against the adversaries of bot detection systems. Our study contributes to the literature of growing study about the new wave of social bots in Twitter, who act in highly deceptive, yet effective coordinated fashion, and have shown evolutionary interaction, and behavioral traits compared to the bots studied previously in the literature. The application of the behavioral traits we have discovered to explore different variety of coordinated Twitter bots, on the wild would be a very interesting

avenue for future work. We also believe converting the experimental inferences of our study to statistical measures, which could possibly be extended to an expert system in detection of social bots, is a major remaining challenge as we look forward to join forces on bringing down these new waves of bots on Twitter.

Chapter 4

Knowledge Bots: GO-Chatbot

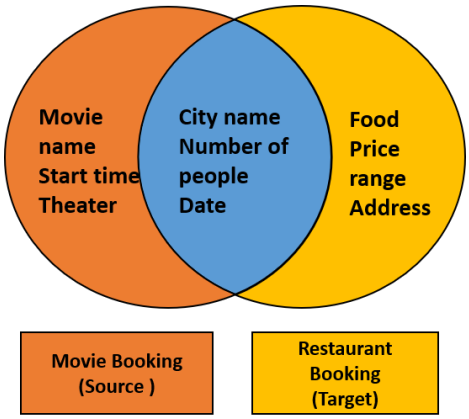
The discussion of this chapter is concentrated on Knowledge bots. Knowledge bots are normally used as conversational agent or chatbot in different web portal or mobile app. Based on the nature of the conversation, the conversational agents or the chatbots can be divided into two categories, such as i) open-domain [132, 145] and ii) closed-domain conversational systems [153]. The closed domain systems have predetermined goals [118] of the conversation. Whereas, for open-domain conversation there is no such goal determined beforehand. An open-domain conversational system can be considered as a combination of multiple closed-domain conversational systems. Conversational agents can be classified into two categories based on their nature of internal operations: i) retrieval based agents and ii) generative agents. Retrieval-based model responds based on predefined responses from input. While the generative agents generate the responses based on previously learned data. Retrieval-based models use conventional rule-based and/or statistical response ranking strategies. In both cases, the knowledge base should contain appropriate dialogue to have a meaningful and engaging conversation. So to build a successful conversation system, knowledge base or dialogue set is the most important prerequisite. As retrieval-based conversational agents mainly use conventional rule-based and/or statistical response ranking strategies, it may not require a huge amount of data. While the generative agents use deep learning techniques to generate dialogues, they need lots of data for training, validation and testing purpose. Access to appropriate dataset (dialogues in this case) is always a big obstacle for any type of deep learning research. Sometimes there is not enough data to develop even retrieval-based agents. So, the success of an open-domain conversational system always depends on how many different closed-domain conversational systems are associated and their accuracy. In this chapter, I am going to focus on closed-domain conversational systems or in other words Goal-Oriented conversational systems or Goal-Oriented chatbots. The success of the chatbot is very much dependent on the quality of its knowledge base. It has been observed that due to inadequate data in the knowledge base of the chatbots, the performance of the chatbots fails in many different ways. This research is directed to find different solutions to this problem. Two leading solutions are discussed in this chapter are: i) use of similar data from other sources using different transfer learning techniques, ii) the use of various generative deep learning networks.

To develop a conversational system, two types of architectures are used such as traditional task-oriented dialogue and fully data-driven dialogue. A typical task-oriented dialogue agent is composed of four modules: (1) a Natural Language Understanding (NLU) module for identifying user intents and extracting associated information; (2) a state tracker for tracking the dialogue state that captures all essential information in the conversation so far; (3) a dialogue policy that selects the next action based on the current state; and (4) a Natural Language Generation (NLG) module for converting agent actions to natural language responses. In recent years, there has been a trend towards developing fully data-driven systems by unifying these modules using a deep neural network that maps the user input to the agent output directly. Most of the implementations of GO chatbots can be considered as a combination of a pipeline modules, where the output of one module is input to the next module. In this system, chatbots always have a big database to answer all the questions [157], [141]. Typically, reinforcement learning (RL) algorithms such as, Deep Q-Nets (DQN) [107] are used to train those types of chatbots. In [90] used RL in their work on "Goal-Oriented Dialogue System." On the other hand, social chatbots use fully data-driven dialogue system, as the main purpose of social bots is to behave like a human being. These types of chatbots normally employ fully supervised, sequence-to-sequence [136] models. They initially encode the user context and request, and later decode an answer. This type of model requires a considerable amount of conversational data to mimic the knowledge of a human [152]. This research is published in [71].

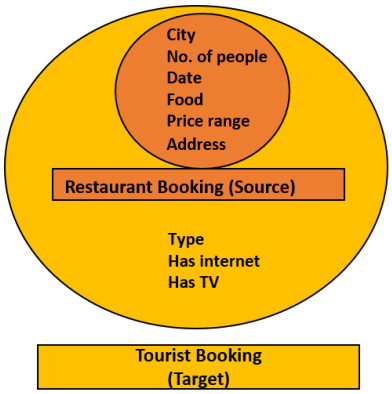
4.1 Motivation

As mentioned previously, to develop a GO chatbot, a sufficient amount of data is required to answer all the possible inquiries. An adequate amount of data is not always available to create a good knowledge base. To resolve the problem [74] came up with a solution to use transfer learning. In [74] a transfer learning method is used to mitigate the effects of low in-domain data availability. In their research, they used restaurant booking and movie booking systems; as both the conversational systems share large extent of common conversation pattern. They mainly addressed two cases: i) when the two domains have an overlap i.e domain overlap and ii) when one domain is an extension of another i.e domain extension.

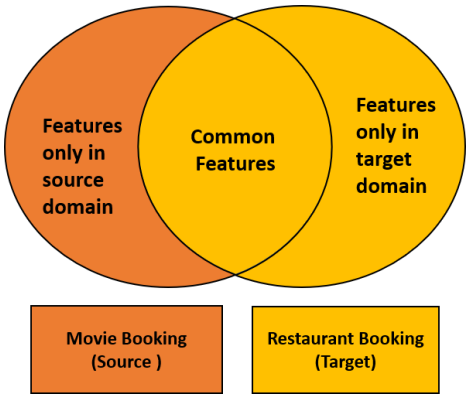
The research of [74] is the motivation of our research. In [74] the authors mainly dealt with domain overlap and domain extension scenarios of the datasets, where the chatbot is answering about the common features of source and target domain. Other than domain extension and domain overlapping, there may occur a third scenario, where the chatbot needs



(a) Domain overlapping



(b) Domain extension



(c) Hybrid domain

Figure 4.1: Diagrammatic representation of three different domains

to answer about the features which are not common between source and target domain. This scenario can be considered as a hybrid scenario. Figure 4.1 is a diagrammatic representation of all the three domains; where Figure 4.1a and Figure 4.1b are representing domain overlapping and domain extension, Figure 4.1c is representing a case called hybrid domain. In the hybrid case, the chatbot of target domain needs to address those features which are

not present in the common feature set of source and target domain. In this article, we have proposed a chatbot system which can answer for all three domain cases. The objective of this paper is to develop a chatbot which can be used for all the three aforementioned scenarios using transfer learning and attention mechanism. An organ recipients' conversation dataset is also introduced in our paper. In section 2 description of datasets is presented, the proposed methodology is described in section 3. The experiment, result and conclusion are described in section 4 and 5 accordingly.

4.2 Dataset

Two datasets are used to perform the entire experiment. The dataset used by [74] is used initially to compare the accuracy of the proposed model with the previous research work. Later on, we developed a second dataset, which exhibits the nature of the hybrid domain. For the rest of this article, we'll refer the dataset of [74] as 'Old datasets' and the second dataset as 'New datasets'.

4.2.1 Old datasets

There are mainly two types of conversational datasets; movie ticket booking data, restaurant booking. These datasets contain some common fields as previously mentioned in domain overlapping or domain extension descriptions.

4.2.2 New datasets

The new dataset [2] is created using the Transplant Candidate Registration Worksheet [45] information. We created data for Transplant Candidate Registration Worksheet information for five different organs such as kidney, lung, heart, pancreas and liver. The data is very similar to the actual conversational data between an organ recipient and a officer in organ transplant center. Initially, we collected all the possible answers for each question. For this experiment, we have generated 100,000 conversational data. While generating the conversational data, the answer for every question is randomly selected from that pool of possible answers. In the conversational data, the patient is looking for two answers from the chatbot: i) what are the nearest organ transplant centers ii) how many patients are waiting for a particular organ in that organ transplant center.

In the Transplant Candidate Registration Worksheet, there are common information inquiries that everyone is required to fill in (e.g. whether the patient has an autoimmune disease or not). On the other hand, there are some specific questions those depend on the type of organ to be transplanted (e.g. for the patient of a heart transplant, whether the patient

has any history of heart attack or not is a very important question). The data for Transplant Candidate Registration Worksheet is used as our new datasets to explore the hybrid domain nature of the data. According to our proposed methodology, while the general clinical information can be learned and transferred from one type of organ to another using transfer learning, organ-specific clinical information can be learned using the attention mechanism.

4.3 Methodology

In this research, we are using the neural dialogue simulator system proposed by [90]. In [90], the researchers used this simulator to build a GO Chatbot in a movie ticket booking system. This framework consists of mainly two major parts i) a user simulator and ii) neural dialogue system. Initially, the proposed model learns the policy using neural dialogue system framework. Once the policy learning process is done, the knowledge can be transferred using the transfer learning technique from the learned source domain to the target domain. Later domain-specific knowledge for the target domain can be learned using the attention mechanism. The schematic diagram of the proposed methodology is presented in Figure 4.2.

4.3.1 Neural Dialogue System

Neural Dialogue System [90] is consisting of five important units which are described in the following subsections.

Language Understanding (LU)

The two most important objectives of LU unit are: automatically identifying the domain (e.g. organ transplant) from a user query and finding the required parameters or slots (e.g. hypertension, heart attack).

Dialogue Management (DM)

The DM unit consists of the Dialogue State Tracker (DST) and the Policy Learning Module, or the agent. DM also uses the knowledgebase to get a suitable reply to the queries. DST tracks the current state and previous states of the dialogue and helps the policy learning module.

Policy Learning

In this section reinforcement learning (RL) is used to learn the pattern questions and respective answers. In the case of RL, policy learning is a major step. The policy helps to select the next system actions, which drives the user to achieve the goal in the minimum number of steps. So, the success of an RL system depends on the goodness is its policy. In the present scenario, we have used Deep Q-Networks (DQN) [143] for policy learning. DQN finds a policy which maximize the reward for the state-action function $Q(s,a|\theta)$. Where s is the state of the agent, θ is latent parameters and the agent is following the policy $\pi = P(a|s)$. DQN stores the experiences (e_t) of the agents in an experience reply buffer $D_t = e_1, ..., e_t$. Agent's experience is expressed as $e_t = (s_t, a_t, r_t, s_{t+1})$. It suggests that if agent at t time is on current state (s_t), takes an action (a_t) then it goes to a new state (a_{t+1}) with a reward (r_t).

In this case, the state of Dialogue State Tracker (DST) (s_t) is the input for the agent. The agent is using ε-greedy policy to take a new action (a_t). So, the agent takes a random action with a probability of ε and the rest of the time it chooses the state giving the maximum award. For every slot, the agent has two options for action: either to ask the user for a constraining value or to suggest to the user value for that slot. Other than that, for opening and closing the dialogue there are two slot-independent actions are there.

If one conversation is over within a predetermined number (n_{max_turns}) of dialogue then the conversation is considered as a successful conversation. The agent receives a positive reward ($r_{positive}$) if it conducts the conversation successfully. Otherwise, it receives a negative reward. There are two types of negative rewards: $r_{negative}$ and $r_{ongoing}$. An agent may receive a $r_{negative}$ negative reward in two different ways. Firstly, if the agent does not answer properly to the user and the conversation ends with a wrong answer from the agent. Secondly, if the conversation does not end within a predetermined number (n_{max_turns}) of dialogues then also the agent receives a negative award ($r_{negative}$). On the other hand, for each of the intermediate dialogue, the agent receives a negative reward ($r_{ongoing}$) .

A sequence of states, actions and rewards, which ends with terminal state is called an episode in RL literature; and one time training a learning machine with all training data is called one epochs. The values of number of episodes ($n_{episodes}$) and number of epochs (n_{epochs}) are also predetermined.

User Simulation

The User Simulator unit creates a user - bot conversation, given the semantic frames. This simulation system is needed as the model is based on Reinforcement Learning. The user goal consists of two different sets of slots: inform slots and request slots.

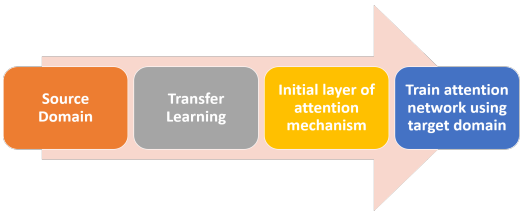


Figure 4.2: Diagram of proposed methodology

Error Model Controller

When training or testing a policy based on semantic frames of user actions, an error model [130] is introduced to simulate noises from the LU component, and noisy communication between the user and the agent in order to test the model robustness. There are two different levels of noises in the error model: one type of errors is at the intent level, another is at the slot level.

4.3.2 Transfer Learning

Using the transfer learning technique, the knowledge of the source domain can be transferred to the target domain. Normally the information in the source domain is much more than the target domain. The primary goal of the transfer learning is to resolve the problem of inadequate data when training a chatbot using an existing similar type of knowledge or data. To transfer the knowledge from the source domain to the target domain, the dialogue state must be modeled in both the domains, and they must share a set of action states. The chatbot in the source domain must be aware of the actions of the target domain even if these actions are never used, vice versa. This requirement stems from the impossibility of reusing the neural weights if the input and output spaces differ. Consequently, when we train the model on the source domain, the state of the dialogue depends not only on the slots that are specific to the source but also on those that only appear in the target one. This insight can be generalized to a plurality of source and target domains and also for the set of actions. So, when training of the target domain, we use the weights of the neural network of the source domain. A diagramatic representation of the transfering knowledge in presented in Figure 4.3.

4.3.3 Attention Mechanism

Attention mechanism [144] is considered as one of the most influential ideas in deep learning. Neural Machine Translation (NMT) [21] is one of the most famous areas where attention mechanism is very successfully used. The attention model consists of one bidirectional long

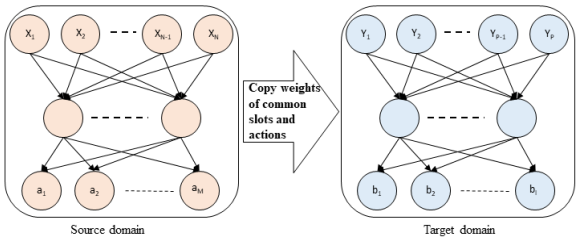


Figure 4.3: Tranfer Learning mechanism used in the research

short term memory (LSTM) [135] layer, one attention layer, one LSTM layer and finally a layer of softmax function which are illustrated in Figure 4.4.

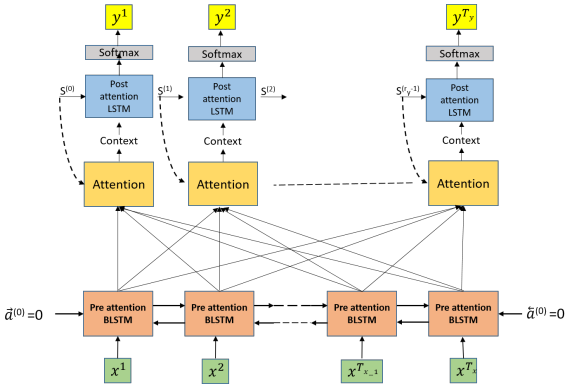


Figure 4.4: Tranfer Learning mechanism used in the research

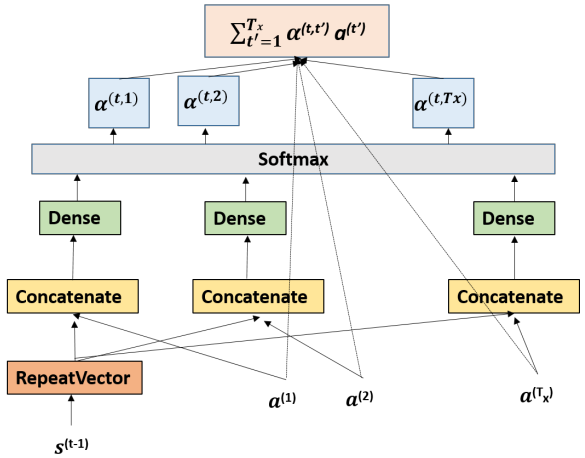


Figure 4.5: Tranfer Learning mechanism used in the research

In this research, the attention mechanism is used to enhance the accuracy of the result of the chatbot, particularly when the chatbot has specific characteristics similar to the hybrid domain. As per our previous discussion, the knowledge of the source domain is transferred

to the target domain; where the initial weights of the target domain are the same as the weights of the source domain after training. In the case of the hybrid domain, after the use of transfer learning, the source domain may not be fully aware of the target domain, especially the domain-related knowledge such as organ-specific clinical information for our new datasets. So, the chatbot may not be capable to handle some specific questions of the target domain properly. To solve this problem, such specific information of the target domain can be trained using the attention mechanism. Therefore, our proposed model can solve problems related to all three scenarios (domain overlapping, domain extension and hybrid domain). In summary, our proposed model initially uses the Neural Dialogue System to learn the policy and train the agent using reinforcement learning on the source domain data. Next, the knowledge of the source domain is transferred to the target domain using transfer. Finally, the target domain related information is learned using the attention mechanism.

4.4 Experiments and Results

We performed two experiments using old datasets and new datasets respectively. The descriptions of those experiment are presented in the following sections.

4.4.1 Experiment with old datasets

In this experiment, the movie booking dataset is used as the source domain and restaurant booking dataset is used as the target domain. While experimenting, we have followed the same steps and parameters used in [74] for the initial learning part and transfer learning to maintain the consistency.

Application of Reinforcement Learning

The reinforcement learning (RL) algorithm is used on the source dataset (movie booking dataset). For the initial policy learning, the RL method used predetermined values of some important parameters such as: Maximal number of allowed dialogue ($n_{dialogues}$), Positive reward ($r_{positive}$), Negative reward ($r_{negative}$), Ongoing dialogue reward ($r_{ongoing}$), Probability of random action taken by agent (ε), Number of training epochs (n_{epochs}), Number of episodes ($n_{episode}$). The values of the are parameters mentioned in the **Value_old** column of Table 5.1. These parameters are used on old datasets.

Application of Transfer Learning

Once the model is trained using the source dataset, the next job is to transfer the knowledge of the source domain to the target domain. In this process, the first step is to identify the

common slots and common actions in the target and source domain. In the next step, the source weights for those slots are copied to the target slots. These transfer weights are going to be the initial weights of the first layer of the attention network. The pseudo-code [74] of this process is mentioned in **Algorithm 1**.

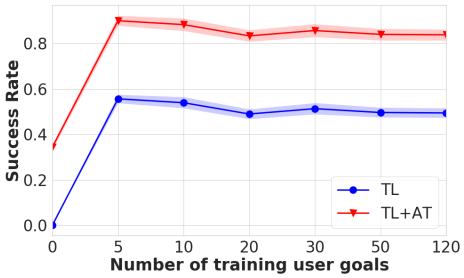
Table 4.1: RL parameters used for the new dataset

Parameter	**Value_old**	**Value_new**
Maximal number of allowed dialogue ($n_{dialogues}$)	20	30
Positive reward ($r_{positive}$)	40	40
Negative reward ($r_{negative}$)	-20	-20
On going dialogue reward ($r_{ongoing}$)	-1	-1
Probability of random action taken by agent (ε)	0.05	0.05
Number of training epochs (n_{epochs})	50	60
Number of episodes ($n_{episode}$)	200	500

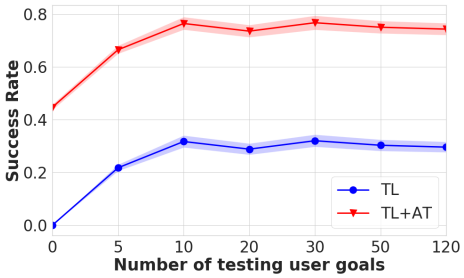
Application of Attention Mechanism

In the last phase of the experiment, the attention mechanism is used which follows the architecture described in Figure 4.4. The attention network is trained using the target domain's data. In the general case, the initial weights of the first layer are randomly chosen, but in our implementation, the initial layer's weights are transferred using the transfer learning method from the source domain.

There are two datasets (movie ticket booking data and restaurant booking data) are present in the category of old datasets. In the experiment, we used restaurant booking as the source domain and movie booking data as the target domain. There is a total of 120 user goals for each of the training and testing data, which are randomly selected into six subsets having 5, 10, 20, 30, 50 and 120 user goals. The experiment is repeated for 100 times to reduce the uncertainty introduced by the random selection. The first model is using only transfer learning (TL) while the second model is using both transfer learning and attention mechanism (TL+AT). The comparisons of performances for training and testing data of TL and TL+AT models on old datasets are presented in Figure 4.6a and 4.6b.



(a) Restaurant booking is source and Movie booking is target domain training performance



(b) Restaurant booking is source and Movie booking is target domain testing performance

Figure 4.6: Performances of TL and TL+AT modles on old datasets

4.4.2 Experiment with New Datasets

In this section, Transplant Candidate Registration Worksheet data for kidney transplant is used as the source domain, while Transplant Candidate Registration Worksheet data for liver, lung, pancreas and heart is used for the target domain. There are 70,000 chats are there for the source domain and 30,000 chats for the target domain.

Application of Reinforcement Learning

In the case of new datasets, Transplant Candidate Registration Worksheet data for kidney transplant is used as the source domain, so initially, the policy learning process in RL is performed using this data. For this experiment, the same set of parameters is used (similar to the experiment on old datasets). However, some parameters have been assigned to different values to get better learning accuracy. The predetermined values for these parameters are mentioned in **Value_new** column of Table 5.1.

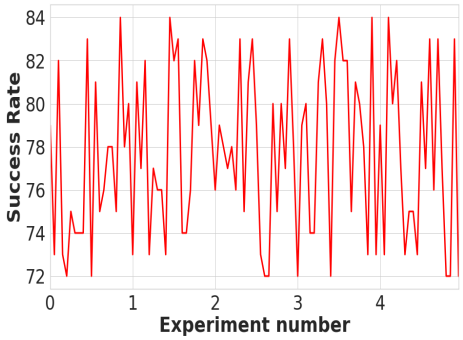


Figure 4.7: Performance of the proposed model using new datasets

Application of Transfer Learning

In this phase Transplant Candidate Registration Worksheet data for kidney transplant is used as source domain while Transplant Candidate Registration Worksheet data for liver, lung, pancreas and heart is used for the target domain. Weights of the common slots and common actions are transferred using a similar technique as mentioned before.

Application of Attention Mechanism

Attention mechanism is used here similar to the old datasets. Here Transplant Candidate Registration Worksheet data for liver, lung, pancreas and heart is used in the attention network.

The experiment is performed 100 times, and on an average, the accuracy is 77.34%. Figure 4.6 demonstrates the plot of the experiment number and its accuracy when applying our proposed methodology on the new datasets.

4.5 Discussion

To verify our proposed methodology, three experiments are performed on two datasets (old and new datasets). Using old datasets two experiments are performed. In the first experiment, transfer of knowledge from the source domain to the target domain is done using transfer learning, and the performance of the chatbot for the target domain is measured. The second experiment has two phases. In the first phase, the knowledge from the source domain is transferred to the target domain using transfer learning, while in the second phase target domain-related knowledge is learned using attention mechanism and finally the performance of the chatbot for the target domain is measured. Our proposed model (transfer learning with attention mechanism) performed better than the baseline method (transfer learning) uisng the old datasets.

For new datasets, the proposed methodology is used and the accuracy of 77.34% is obtained. Using the new datasets, we did not perform any test using only transfer learning, because the new datasets contain mainly hybrid domain data as mentioned in the 4.1 Motivation section hence, only transfer learning can not answer target domain related questions.

There is a scope to improve the performance of the GO chatbot using the new datasets. This can be done using more sophisticated learning algorithms. This research is more focused on proposing a model which can handle all three scenarios of transferring knowledge from one domain to another (domain overlap, domain extension, hybrid domain).

4.6 Conclusion

To find a better solution for the GO chatbots, in this research we proposed a model which consist of three steps: first, using reinforcement learning to learn the policy, secondly using transfer learning to transfer the common knowledge from one domain to another domain and finally using attention mechanism to train the domain-specific knowledge. While transfer learning gives a solution to the problem of inadequate domain-related data, attention mechanism helps to learn the domain-related knowledge. The contributions of this article are primarily, proposing a solution for the hybrid domain of GO chatbots and secondly, introducing a new chatbot dataset for Transplant Candidate Registration of heart, kidney, lung, pancreas and liver. In the future, this research can be extended to achieve a more human-like conversation.

Chapter 5

Knowledge Base Generation Using Generative Adversarial Networks

"Generative Adversarial Networks is the most interesting idea in the last 10 years in Machine Learning." — Yann LeCun, Chief AI scientist at Facebook.

Research on Generative Adversarial Networks (GAN) [59] is producing very promising results, and it is rapidly changing its domain. GAN is being used to generate different forms of data. GAN is most popular among different computer vision applications [51, 86, 92, 101, 113, 146, 154, 166]. GAN is extensively used in image generation [35, 81, 122, 150, 162], image completion [34, 46, 91, 156, 158], image super-resolution [48, 87, 103, 104, 151] and image to image translation [36, 75, 83, 96, 102, 139, 167, 168]. Wide range of applications of GAN in computer vision motivates the researchers to use GAN for different types of data. As a result, recently GAN is also producing very good results for Natural Language Processing (NLP) [42, 53, 77, 156], Time Series Synthesis [31, 47, 52, 68, 89], Semantic Segmentation [48, 99, 123, 134] etc.

Generating synthetic text data is one of the most important most popular applications in NLP. By the advancement of Deep Learning techniques, different Recurrent Neural Networks [26] are used for developing sequence to sequence model [109] to generate synthetic text data. Recent success of GAN for computer vision applications grabbed the attention of the NLP research community to use GAN in different NLP applications. This research is published in [70].

GAN consists of two neural networks: a generator network $(G(\cdot))$ and a discriminator network $(D(\cdot))$ and, they are adversarial to each other. The generator is responsible for generating synthetic data, and the discriminator is a pretrained model able to classify synthetic or real data. The objective of the generator network is to generate some synthetic data which can not be identified by the discriminator as synthetic. GAN is based on the zero-sum non-cooperative game or minimax game. According to the game theory, the GAN model converges when the discriminator and the generator reach a Nash equilibrium. The discriminator is trying to maximize the reward function while the generator is trying to minimize the objective function $V(D,G)$.

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data(x)}}[logD(x)] + \mathbb{E}_{z\sim p_{z(z)}}[log(1-D(G(z)))] \tag{5.1}$$

G is Generator
D is Discriminator
$p_{data(x)}$ is distribution of real data
$p(z)$ is distribution of generator
x is sample data from $p_{data(x)}$
z is sample from $p(z)$
$D(x)$ is Discriminator network
$G(z)$ is Generator network

It is clear from the equation 5.1 that the discriminator is always trying to maximize the probability that real data is identified as real and synthetic data is identified as synthetic; on the other hand, the generator is always trying to minimize the probability and trying to make fool the discriminator. This model is applicable for image generation. In case of image generation, $G(z)$ generates an image and discriminator $D(x)$ classifies it either real of fake or synthetic.

In case of text generation, at Recurrent Neural Network (RNN) is normally used. For RNN the input at h^t hidden step is the output of the previous hidden step $h^{(t-1)}$.

$$h^t = RNN(h^{t-1}, w^{t-1}) \tag{5.2}$$

The hidden state is normally passed through one or multiple liner layers, a softmax [50] layer and a argmax to generate the word. The RNN is trained to generate the next word in every step. In the forward propagation of training process, the RNN picks a word with the highest probability of softmax function's output. The word then compared with the expected output and cross-entropy loss [73] is calculated. Like any other neural network, this cross-entropy loss is back propagated and updates the weights and biases.

Now if we try to use RNN as the generator in GAN, the objective of the training process will be to minimize the value of $(1-D(G(z)))$. we need to feed the output of the generator to the discriminator and back-propagate the corresponding loss of the discriminator. For these gradients to reach the generator, they have to go through the non-differentiable "picking" operation at the output of the generator. This is problematic as back-propagation relies on the differentiability of all the layers in the network. Hence, we can not use GAN to generate text as we have used it to generate images. This is one of major challenges to generate text data using GAN.

Other than the problem of differentiability, GAN also suffers from two other problems: training instability and mode dropping. Unlike image generation, length of generated text varies based on the applications. The loss computed by the discriminator once the entire sentence is generated this makes the entire system very unstable, and the problem becomes critical for longer sentences. The mode dropping problem appears when a particular pattern appears in training rarely. For example, if a GAN is generating images of Trees, all the different kind of trees have some degree of similarities. In case of text generation, some complex text format may appear rarely and the model may not be able to generate that pattern. Out of these three problems, most of the research is done to solve the first problem. In previous research we have seen three major strategies are followed to resolve the problem: i) Reinforcement Learning (RL) based solutions ii) approximation of softmax function iii)making output space of generator continuous. In most of the previous research work, the researchers mainly focused on different variety of the generator. So in the next section in our discussion about previous works of text generation using GAN will be focused on different types of generator network.

5.1 Methods to Generate Text Data using GAN

5.1.1 Reinforcement Learning (RL) based solutions

To generate text data Reinforcement Learning (RL) based technique was first time used by Yu et al. [159]. In this paper, the authors proposed a sequence generation framework called SeqGAN. SeqGAN uses stochastic policy for modeling the generator function in RL. It overcomes the differentiability by using a gradient policy update method. The intermediate state-action steps gets the reward using Monte Carlo search while the complete sequence is judged by the GAN discriminator. In SeqGAN recurrent neural network (RNN) (Hochreiter and Schmidhuber 1997) is used as the generative model. The RNN takes the input word sequence in an embedded format and maps to a hidden layer, and in the output layer is having softmax activation function to produce the final output from the generative layer. Convolution Neural Network (CNN) is used as the discriminative model in SeqGAN. Once the generative model creates the entire sentence of text data, then the discriminative model uses CNN to classify the generated sequence into two classes such as human-generated and machine-generated.

Following the works of SeqGAN, William et al. proposed MaskGAN in [53]. This research work addressed all the three problems of GAN as mentioned earlier in this

discussion. In MaskGAN the researchers proposed actor-critic conditioning GANs to generate text data using GANs. The model tries to predict missing text based on the context. The authors claimed this model can produce more realistic data compared to similar models like maximum likelihood trained model. Like any other GAN this model also has two main components: Generator and Discriminator. The generator and discriminator both have identical structure. They consist of an encoder and a decoder unit.

Let, the input is represented by x and output is represented by y. The input is a sequence of words $X = (x_1, .., x_T)$. Once the input is available, the model generates a binary mask for the input sequence. The length of the mask and the input sequence is equal. The mask m can be represented as $m = (m_1, m_2, .., m_T)$, where $m_t \in 0, 1$. The mask generation process can be deterministic or stochastic, that depends on the implementation strategies. Once the mask is generated the words or the tokens of the sequence are replaced by a special mask token $< m >$ if the value of mask of that position is zero, otherwise it remains unchanged. In other words, if m_t is 0 then x_t will be replaced by $< m >$ otherwise it will be unchanged. In the next step, the masked input sequence $(m(x))$ is used as the input to the encoder. The encoder provides access to future context for the MaskGAN during decoding. Although, the decoder generates the missing token in an auto-regressive process like any other standard language-modeling system, there is an important modification observed in the proposed model. In case of standard language-model the prediction of the next word is done conditioned on the previously generated sequence. In this case it has been done by conditioned on both the masked text $m(x)$ as well as what it has filled-in up to that point.

$$\begin{aligned} P(\hat{x}_1, ..., \hat{x}_T | m(x)) &= \prod_{t=1}^{T} P(\hat{x}_t | \hat{x}_1, ..., \hat{x}_{t-1}, m(x)) \\ G(x_t) &= P(\hat{x}_t | \hat{x}_1, ..., \hat{x}_{t-1}, m(x)) \end{aligned} \tag{5.3}$$

On the other hand, the discriminator is initially trained with the masked sequence and then sequence generated by the generator is given as an input to decode. The model is not fully-differentiable as the process of selecting the next token in the generator part follows some discrete sampling technique. Hence in the training process of the generator, it uses policy gradient method of RL. Policy gradient method is used in [159] for the first time. In MaskGAN the authors used one of the REINFORCE family of algorithms to find the unbiased and estimator.

Other than SeqGAN and MaskGAN, RankGAN [79] is another variation of GAN used for text generation using RL. In RankGAN the author introduced a GAN which is one of the first generative adversarial networks which learns by relative ranking information.

RankGAN follows the basic structure of a GAN and consists of a generator unit and a discriminator unit. Moreover, there is a ranker unit. The ranker unit plays a very important role in RankGAN. The generator generates multiple sentences for a noise input and the ranker ranks all the machine generated and human generated sentences together. The objective of the generator is to generate one sentence that gets a higher rank than the referenced human generated sentence. In this model the rank score is an important function. In this paper cosine similarity is used to find the similarity between machine generated sentence and the reference sentence. To rank in a sentence with respect to a set of sentences a softmax-like formula 5.4 is used, where γ is an empirical constant. In the learning process, the reference sentences are randomly selected from a set of human-written sentences. In this 5.4 rank score is calculated for sequence s and the sequence is compared with C.

$$P(s|U,C) = \frac{exp(\gamma\alpha(s|u))}{\sum_{s' \in C} exp(\gamma\alpha(s'|u))} \tag{5.4}$$

5.1.2 Continuous Approximation of Softmax

The RL methods of text generation using GAN always take long training time, and there is a chance of reaching local optima in those methods. So, instead of using RL, many researchers focus on make a proposing a continuous approximation of the softmax function.

Matt and José in proposed a method to solve the problem of discrete values in GAN using Gumbel-softmax distribution. Gumbel-softmax [85] creates continuous approximation of softmax function.

Gumbel-softmax distribution Let p is a d dimensional vector, contains probabilities of multinomial distribution on y with $p_i = p(y_i = 1), i = 1,2,..,d$. h is vector representation of y in a continuous d-dimensional form and g is independent and follow a Gumbel distribution with zero location and unit scale.

$$p = softmax(h) \tag{5.5}$$

It can be shown that sampling y according to the previous multinomial distribution with probability vector given by 5.5 is the same as sampling y according to

$$y = one\_hot(argmax(h_i + g_i)) \tag{5.6}$$

The sample generated in 5.6 has gradient zero with respect to h because the one_hot(arg max(·)) operator is not differentiable. This is approximated with a differentiable function based on the soft-max transformation [8]. In particular, we approximate y with

$$y = softmax(\frac{1}{\tau(h+g)}) \tag{5.7}$$

where τ is an inverse temperature parameter. When $\tau \to 0$, the samples generated by 5.7 have the same distribution as those generated by 5.6 and when $\tau \to$ inf, the samples are always the uniform probability vector. For positive and finite values of τ the samples generated by 5.7 are smooth and differentiable with respect to h.

In [85] Matt and José used Gumbel-softmax distribution to generated text data using GAN. In the implementation of GAN they used Long Short Term Memory (LSTM) network for both generator and discriminator module.

Yizhe et al. [164] proposed an feature matching technique to generate text using GAN. In this model, a Soft-argmax approximation technique [163] is used instead of the traditional softmax function. In the discriminator module, features are extracted form both the real (f) and synthetic data ($\tilde{f}$) and Maximum Mean Discrepancy (MMD) [61] is measured between empirical distribution of sentence embeddings f and $\tilde{f}$. It has been found this model is difficult to train in practice. Specifically, (i) the bandwidth of the RBF kernel is difficult to choose; (ii) kernel methods often suffer from poor scaling; and (iii) empirically, TextGAN tends to generate short sentences. To solve these problems Liqun et al. proposed a text generation technique using Feature-Mover's Distance or Feature Mover GAN (FM-GAN) [33]. In discriminator the the Earth-Mover's Distance (EMD) is used to find the difference between the sentence features of real and synthetic data.

5.1.3 Making output space of generator continuous

To generate text data using GAN different autoencoders and modified sequence to sequence models are also used. TextKD-GAN [64] model used autoencoders to generate text data using GANs. Here autoencoder is used to create a continuous representation of sentences, which is a smooth representation that assign non-zero probabilities to more than one word. This smooth representation of the input is used to train the generator to generate similar types of smooth representations. The discriminator will get the input as a continuous representation of the generator which makes the job difficult for the discriminator compared to one-hot input as stated in IWG [62]. Adversarial Text Generation Without Reinforcement Learning: LaTextGAN [47] (latent-space GAN for text) is proposed by David Donahue and Anna Rumshisky. This model also used an Autoencoder module to solve the problem of discrete representation of text data in GAN. The entire operation is divided into two stages: i) training the AE module ii) training the Generator and Discriminator module. In

the first step the AE unit is trained using all the data available. Then some sample noise is generated and the noise is used as the input for both the Generator and Encoder unit. Both Generator and Encoder units generate some data which is the input for the Discriminator unit. In this paper the researcher used IWGAN using the equation 5.8 for Generator(g_θ) and Discriminator (f_w).

$$\max_{\theta} E_{z\ p(z)}[f_w(g_\theta(z))] - E_{x\ p(x)}[f_w(x)] \tag{5.8}$$

In [100] Oswaldo Ludwig proposed a model for generative conversational agents (GCA). The objective of this research is to generate dialogues, which is very similar to present research objective. So in [100] has been explored further and the GAN is proposed by Oswaldo Ludwig is used to develop a Self-Attention Generative Adversarial Network (SAGAN) [161] to generate dialogues more efficiently.

This study has explored two [100] and 5.8 of the above mentioned methods of text data generation using GAN for further enhancements. As per the objective of the study the GAN has been used for generating dialogues in both the cases.

In section 5.2 both the proposed methods are described; in section 5.2 the methodology is presented; description of data and experiment is presented in section 5.2.4 and 5.2.5; result and conclusion are described in sections 5.3 and 5.5 respectively.

5.2 Proposed Methods

Text generation using GAN is giving very promising results, which is why exploring the different possibilities of GAN in conversational dialogue generation is the primary motivation of this research work. The recent development of SAGAN gives us the ability to track tiny details of a training dataset for image or computer vision research, but this new methodology has not been used for dialogue generation; this very fact propels our motivation to focus on this research. In SAGAN, self-attention mechanism is used, which normally produces a great result in NMT because of its inherent capability to track the context of a sentence. In this research, the self-attention network is used very effectively to maintain the context of the dialogue. In our knowledge, this is the first research where SAGAN is used for dialogue generation, which is the most significant contribution of the research work. To implement SAGAN for text data generation GCA and LaTextGAN are used as underline GANs. In the rest of this article, the implementation of SAGAN using GCA is referred to as method 1 and the implementation of SAGAN using LaTextGAN is referred to as method 2.

There are two significant components of SAGAN: i) GAN and ii) Self-attention unit.

Self-attention is the standard procedure which is followed in both the methods and implementation of GAN is different in both the approaches. In this section, the general workflow of a SAGAN system is described initially, and then the workflow of self-attention model, GCA, and LaTextGAN are described. The diagrammatic representation of the generic structure of the SAGAN followed for both the methods is presented in Fig. 5.4. This methodology is similar to SAGAN, but implementation is different as here target is to generate the text data. The generator is following the GCA or LaTextGAN algorithm and the discriminator is determining the quality of generated dialogue.

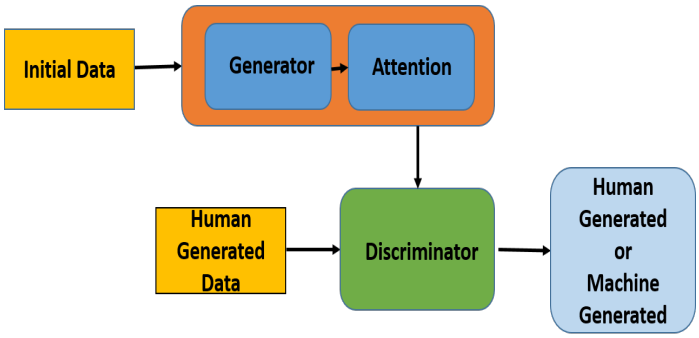


Figure 5.1: Proposed SAGAN for dialogue generation

5.2.1 Self-Attention Mechanism

Attention mechanism [144] is considered as one of the most influential ideas in deep learning. Neural Machine Translation (NMT) [21] is one of the most famous areas where attention mechanism is very successfully used. The attention model consists of one bidirectional long short term memory (BLSTM) [165] layer, one attention layer, one LSTM layer and finally a layer of softmax function which are illustrated in Fig. 4.4.

In SAGAN the self-attention mechanism is used for modeling long-range dependencies and keeping track the details of images. In the SAGAN, the proposed attention module has been applied to both the generator and the discriminator, which are trained in an alternating fashion by minimizing the hinge version of the adversarial loss [93], [140], [106]. While applying self-attention mechanism in SAGAN, the features from the previously hidden layer as input for both generator and the discriminator. In Fig. 5.2 different layers of attention network is shown. The input vectors are basically outputs of the previus generator and the discriminator. As per Fig. 5.2, $x^{(i)}$ is the i^{th} input to the self-attention network and the first layer is a bi-directional RNN. The forward and backward activation functions are

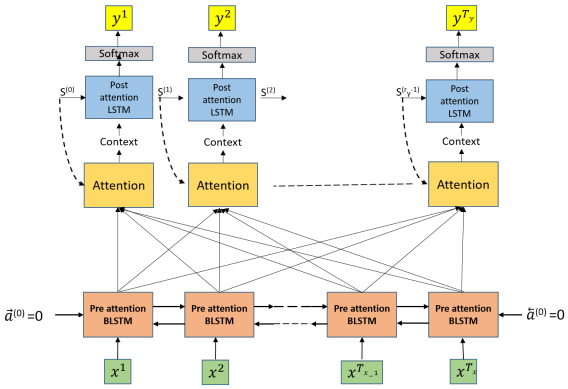


Figure 5.2: Tranfer Learning mechanism used in the research

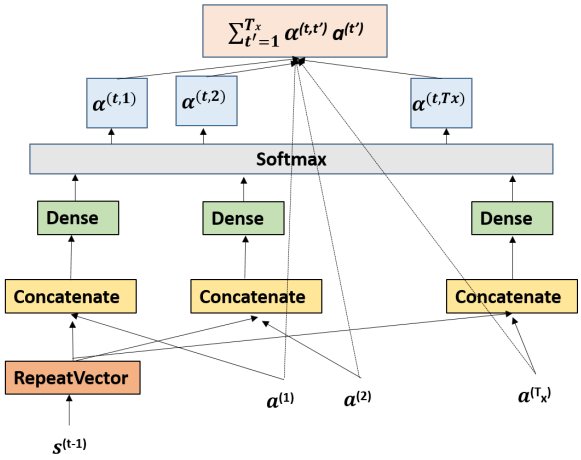


Figure 5.3: Tranfer Learning mechanism used in the research

represented as $\vec{a}^{<t>}$ and $\overleftarrow{a}^{<t>}$. Features from forward and backward RNN is concatenated together and represented as $a^{<t>} = (\vec{a}^{<t>}, \overleftarrow{a}^{<t>})$. The next leyer is a single dimension RNN represented by $s^{<t>}$ and the input to this layer is context (c) and the output is our desired result. The value if c depends on the attention parameters $\alpha^{<1,1>}, \alpha^{<1,2>}, ...$, tell us how much the context depend on the features. The context is the weighted sum of the features, which is shown in Fig. 5.3. So, $\alpha^{<t,t'>}$ is the amount of 'attention' $y^{<t>}$ should pay to $a^{<t'>}$.

$$\sum_{t'} \alpha^{<t,t'>} = 1 \tag{5.9}$$

$$c^{<t>} = \sum_{t'} \alpha^{<t,t'>} a^{t'} \tag{5.10}$$

The attention weight $\alpha^{<t,t'>}$ is calculate using the following formula:

$$\alpha^{<t,t'>} = \frac{exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} exp(e^{<t,t'>})} \tag{5.11}$$

To calculate the factor $e^{<t,t'>}$, a small neural network is used, where $s^{<t-1>}$ is the neural network state in the previous time stamp. The other input is $a^{<t'>}$ the feature from the time stamp t'.

5.2.2 Generative for Conversational Agent (GCA)

In GCA [100] the generator is using greedy decoding technique to generate data. The GCA with a greedy algorithm is presented in Algorithm 1.

Input: **x**: the input sequences (context text)

Output: **y**, p: the sampled output sequence and its conditional probability

$p(\mathbf{y}|\mathbf{x})$

$p \leftarrow 1$

$\mathbf{y} \leftarrow []$

$y \leftarrow$ 'BOS' (symbol representing the beginning of the sentence);

while $y <>$ 'EOS' (symbol representing the end of sentence) **do**

  $\mathbf{y} \leftarrow [\mathbf{y},y]$

  input **x** and **y** into the two input layers of the model

  $y \leftarrow$ token corresponding to the largest output of the model

  $p(y\,|\mathbf{x},\mathbf{y})$ the value of the largest output of the model

  $p \leftarrow p \times p(y\,|\mathbf{x},\mathbf{y})$

end while

Initially, the size of all sequence are not the same, but in the data preprocessing stage using the zero-padding technique, the size of all sequence are made S_s. After that, each

of the sequence vectors is encoded into a one-hot vector and each sequence is represented as $\bar{x_i} \in \mathbb{R}^{S_v}$. S_v is the size of the vocabulary. The incomplete answers are represented by $y \in \mathbb{R}^{S_s}$ and in the preprocessing stage, they are also converted into a one-hot vector. The one-hot vector representation of incomplete answers are represented as $Y = [\bar{y_1}\bar{y_2}...\bar{y_{s_s}}]$.

The GCA architecture basically models $P(y|x)$, where dialogue history is represented by $x \in \mathbb{R}^{S_s}$. $\mathbb{R}^{S_s}$ is the sequence of token indices and the incomplete answers are represented by $y \in \mathbb{R}^{S_s}$. GCA represents the $P(y|x)$ in the following manner:

$$P_\theta(y|g(x)) = \prod_{i=1}^{s_s} P_\theta\left(y_i|f_\beta(y_0...y_{i-1}), g(x)\right) \tag{5.12}$$

End-to-end Adversarial Training by Backpropagation

Adversarial training requires a human-generated dialogue dataset ($\mathcal{H}$), a generator (G) and a discriminator (D). The primary objective of the generator is to generate artificial dialogue dataset in such a way that discriminator can not find the difference between a machine-generated dataset and human-generated dataset. In this method, the discriminator performs token level binary classification; this is why, after generation of each token, the discriminator classifies it either human-generated or machine-generated. To do so, the discriminator takes input either generated by the generator or utterances from the human-generated dataset. If the input comes from G, then it is represented by y^- otherwise ($\mathcal{H}$) it is represented by y^+. The input vectors to D are preprocessed and generate a dense matrix using one embedded matrix ($W_e \in \mathbb{R}^{s_e \times s_v}$), s_e is an arbitrary dimension in word embedding vector. Two dense vectors can be represented as, $E_c \in \mathbb{R}^{s_e \times s_v}$ and $E_a \in \mathbb{R}^{s_e \times s_s}$. In Fig 5.4. a flowchart of the proposed methodology for SAGAN using GCA is presented, where application of different deep learning algorithms in different phases are also described.

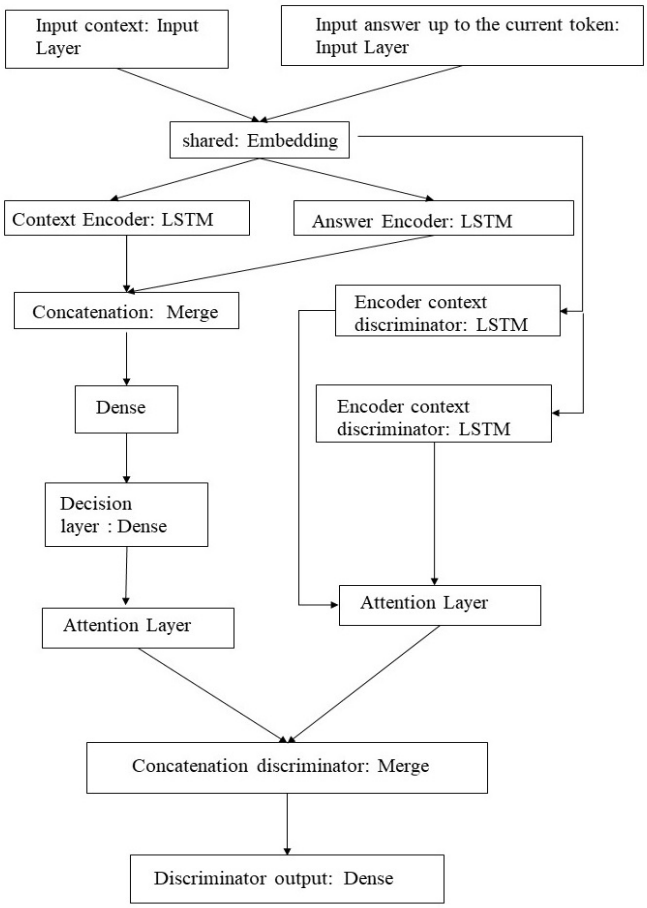


Figure 5.4: Flowchat of the proposed SAGAN using GCA

$$E_c = W_e X \tag{5.13}$$

$$E_a = W_e Y \tag{5.14}$$

Now both E_c and E_a are processed by two LSTMs, Γ_{cd} and Γ_{ad} respectively. Γ_{cd} encodes the previous utterances or the context and Γ_{ad} encodes the complete answer up to the current token and yields embedding vectors of two sentence.

$$e_{cd} = \Gamma_{cd}(E_c; \mathcal{W}_{cd}) \tag{5.15}$$

$$e_{ad} = \Gamma_{ad}(E_a; \mathcal{W}_{ad}) \tag{5.16}$$

$\mathcal{W}_{ad}$ and $\mathcal{W}_{cd}$ are LSTM parameters of D. In the later stage, the vectors are concatenated with the generator output (p) and provided to a dense layer with sigmoid activation function that outputs $l \in [0,1]$, with 1 corresponding to a perfect match with the class human-generated and 0 to the class machine-generated. The dense layer is defined as the following:

$$e_d = [p \;\; e_{cd} \;\; e_{ad}] \tag{5.17}$$

$$l = \alpha(\mathcal{W}_d e_d + b_d) \tag{5.18}$$

In equation 7, $\mathcal{W}_d$ is representing the weights and b_d is representing the bias vector and $\alpha(.)$ is the sigmoid activation function.

5.2.3 SAGAN using LaTextGAN

The second method of SAGAN implementation is based on the GAN architecture proposed in LaTextGAN. In this case the GAN is divided into two stages, i) encoder and decoder section, ii) generator and discriminator section.

Encoder and Decoder: To overcome the problem of discrete text generation using GAN encoder and decoder models are used, it provides a continuous output space for the generator. The training data is first trained with this encoder and decoder model. The output of the encoder is the continuous output space we are looking for the generator to use in the later stage. In the proposed model the structure of encode and decode is enhanced with the addition of attention layer as mentioned in the earlier section.

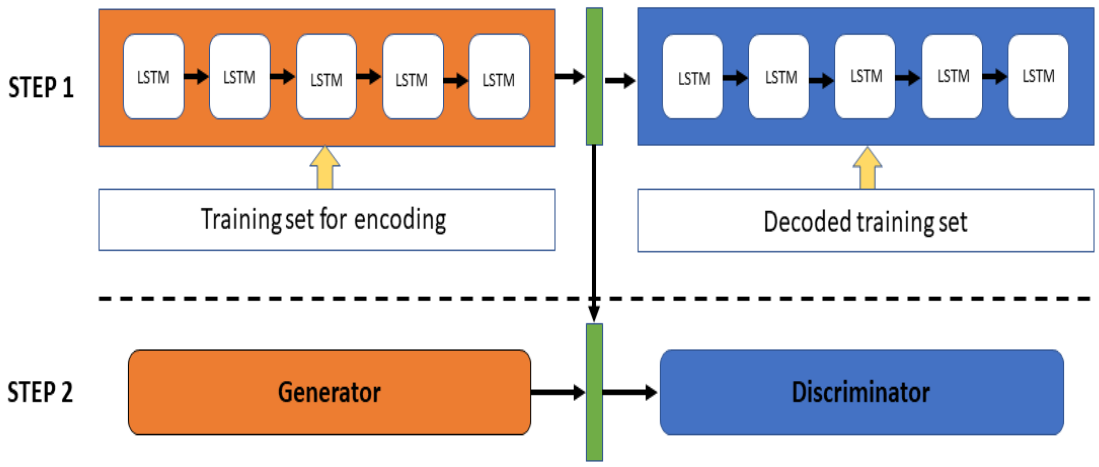


Figure 5.5: Flowchat of the proposed SAGAN using GCA

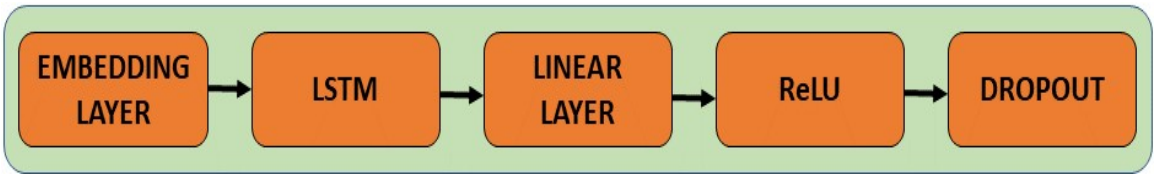


Figure 5.6: Encoder of the proposed SAGAN using GCA

Figure 5.7: Decoder of the proposed SAGAN using GCA

Generator: Once the encoder and decoder module is adequately trained, then the generator starts working. The generator module of the SAGAN is a fully-connected network. It consists of a few blocks of working units. Each of the blocks is also a small network of three layers. The first and the layer in a block is a liner layer, and there is a RaLU function layer present in between these two layers.

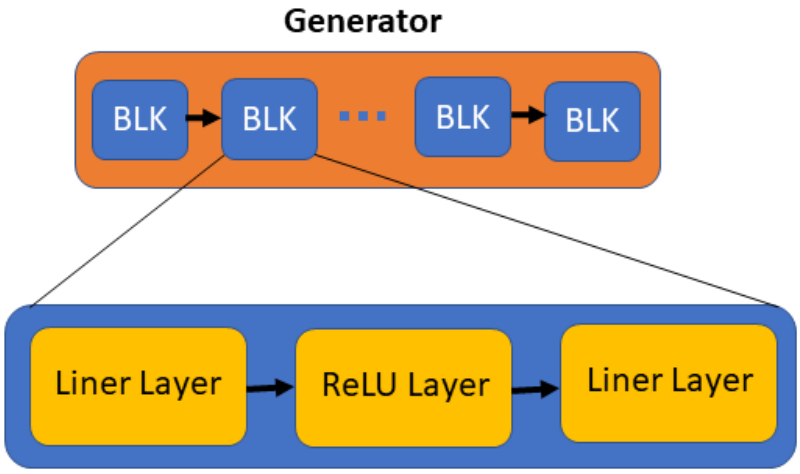


Figure 5.8: Schematic Diagram of Generator

In the generator module, a random number is generated initially, then it goes through the generator module, and finally, the decoder decodes it. The decoded output is then evaluated by the discriminator unit to find out the loss. In this process, the goal is to train the generator in such a way, that for any random number it can generate values which are from the same distribution of the output of the generator. If the generator can do it successfully, then the decoder can decode the values into text data which belong to the same domain of the desired output.

Discriminator: In the proposed SAGAN using LaTextGAN method, gradient penalty is used a the discriminator function. This penalty function was introduced by Ishaan Gulrajani1 et al. [62] to improve the training of Wasserstein GANs. The penalty function is represented is Equation 5.19

$$L = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] + \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] + \lambda\mathbb{E}_{\hat{x}}[(\|(\nabla_{\hat{x}}D(\hat{x}))\|_2 - 1)^2] \tag{5.19}$$

In the gradient penalty function consists of two types of errors, the traditional discriminator error and newly added gradient penalty. In the 5.19 the first half is representing the critic loss or the traditional loss in GAN, while the second part is representing the gradient penalty. In the equation, $\mathbb{P}_r$ and $\mathbb{P}_g$ are representing data distribution and model distribution respectively, implicitly defined by $\tilde{x} = G(z)$. Where z is the input to the generator, z is sampled from a noise distribution p (uniform distribution or a spherical Gaussian distribution). Gradient penalty is introduce to implement a soft version of the constraint with a penalty on the gradient norm for random samples $\hat{x} \sim \mathbb{P}_{\hat{x}}$. To calculate the gradient penalty Algorithm 2 is

followed in [62]. To execute this algorithm a few predefined parameters are required. 1) The coefficient of the gradient penalty ($\lambda = 10$), 2) Critic iteration number for each generator ($\eta_{critic} = 5$), 3) Three parameters for Adam optimizer [137] ($\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$), 4) Batch size (m).

Algorithm 2

Require: Initial critic parameters w_0, initial generator parameters θ_0

while θ has not converged **do**

  for t = 1 **to** η_{critic} **do**

    for i = 1 **to** m **do**

      Sample real data $x \sim \mathbb{P}_r$, latent variable $z \sim p(z)$, a random number $\varepsilon \in U[0,1]$

      $\tilde{x} \leftarrow G_\theta(z)$

      $\hat{x} \leftarrow \varepsilon x + (1-\varepsilon)\tilde{x}$

      $L^{(i)} \leftarrow D_w(\tilde{x}) - D_w(x) + \lambda \mathbb{E}_{\hat{x}}[(\|(\nabla_{\hat{x}} D(\hat{x}))\|_2 - 1)^2]$

    end for

    $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2)$

  end for

  Sample a batch of latent variables $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$

  $\theta \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2)$

end while

5.2.4 Dataset

In this experiment two datasets are used the first dataset is provided by [100]. The result of [100] is considered as baseline for this research, so the same dataset is used.

Dataset 1: In the case of the first dataset, the human-generated dialogues are collected from the online English course, because it has been observed that the dialogues are grammatically correct. Since the training set is grammatically correct the generated text will follow the sentence structure and it is expected to produce grammatically correct sentences. This dataset consists of 6921 single-turn conversations. A single-turn conversation consists of one question and its answer. The dataset is divided into training (80%) set and testing set (20%) and used for the experiment.

Dataset 2: The second dataset or UDC is extracted from the Ubuntu Relay Chat Channel [133]. The dataset consists of 1.85 million conversations and on an average, there are five utterances in each conversation. The entire UDC is divided into training, validation and test sets (90%, 5%, 5%). The total number of unique words in the corpus is limited to 50,000. All other words are marked as 'unknown' (UNK).

Data Preprocessing

The data preprocessing steps are followed for both the datasets. For data preprocessing python's NLTK [23] library is used.

A. Only English words are considered for this experiment. From the second dataset some Chinese words are remove.

B. All English words are converted to lower case letters using NLTK.

C. In the next step all the abbreviated verbs (like: won't, won 't, 'wouldn't, wouldnt́, 'm) are replaced by their full verb forms (will not, will not, would not, would not, am).

D. Both the datasets contain lots of special characters (-, _, *, / etc) to reduce the complexity of the computational job these special characters are removed using some regular expression.

E. Once the above steps are performed for each dataset, word-index and index-word dictionaries are created to represent the words in terms of numerical values.

F. Then the sentences are represented in one-hot vector form and finally in an embedded format to feed into the proposed model.

5.2.5 Model Implementation Details

Once the preprocessing steps are done, both the datasets are feed into to the two proposed SAGAN models, GCA and LaTextGAN model. The performances of both the SAGAN models are compared with their respective GAN models. For both the SAGAN models, the respective GAN models are considered as the baseline for the performance comparison. The models are implemented using Python 3.7, Scikit Learn [117], Keras [37] and PyTorch framework [115]. For implementation, there are some hyperparameters used to tune the model and achieve good results. The hyperparameters used for the GCA model and the proposed SAGAN model based on GCA for Dataset 1 and Dataset 2 are given in Table 5.1. All the common parameters are same for both the models; the Self-attention module is implemented as described in Section 5.2. The hyperparameters used for the GCA model, and the SAGAN model based on CAG are presented in Table 5.1. Similarly, the hyperparameters used for the LaTextGAN model, and the SAGAN model based on LaTextGAN are presented in Table 5.2.

Table 5.1: HYPER-PARAMETERS USED FOR THE GCA MODEL AND SAGAN USING GCA MODEL

Hyper-parameters	Base model values	Proposed model values
Word embedding size	100	100
Sentence embedding size	100	100
Maximum input length	50	50
Maximum output length	50	50
Number of LSTM layers	2	2
Decoder size	3500	3500
Epochs	4	4
Batch size	128	128
Dropout	0.25	0.25
Learning rate G (α_g)	$5e^{-5}$	$5e^{-5}$
Learning rate D (α_d)	$1e^{-4}$	$1e^{-4}$
Number of B-LSTM layers	NA	1

Table 5.2: HYPER-PARAMETERS FOR LATEXTGAN MODEL, SAGAN USING LATEXTGAN MODEL

Hyper-parameters	Base model values	Proposed model values
Batch size	32	32
Maximum Sequence length	20	20
Learning rate	0.0005	0.0005
Autoencoder		
Word embedding size	200	200
Encoder hidden dimension	100	100
Latent dimension	100	100
Dropout	0.5	0.5
Decoder hidden dimension	600	600
Number of layers	20	20
First Liner layer dimension	100x100	100x100
Activation function	ReLU	ReLU
Second Liner layer dimension	100x100	100x100
α	0.0001	0.0001
β_1	0	0
β_2	0.9	0.9
λ	10	10
Number of layers	5	5
Attention layer	NA	Yes

5.3 Result

Comparing the results of different models is the last phase of this research work. The dialogues generated by the four different GAN models are compared based on a particular evaluation function. In the previous research, the researchers used [47] bilingual evaluation understudy score or BLEU score [114] as the evaluation score for the text generation. Hence, to compare the results with previous results, the BLEU score is used as the evaluation score for this research work also.

BLEU Score: As the name suggests BLEU score is mainly used to evaluate the performance of the neural machine translation (NMT) [114] jobs. In the case of NMT, the generated sentence is compared with one or multiple expected sentences. The BLEU score is calculated using the Equation 5.3.

$$P_n = \frac{\sum_{n\_gram} Countclip(n\_gram)}{\sum_{n\_gram} Count(n\_gram)} \tag{5.20}$$

The numerator is the total number of n-gram sequences present in the reference sentences, and the denominator is all the possible n-gram sequences of the reference sentences. In this research, bi-gram sequences are used for BLEU score calculation. An example of BLEU score calculation is presented in Table 5.3.

Reference 1: The cat is on the mat.

Reference 2: There is a cat on the mat.

NMT output: The cat the cat on the mat.

Table 5.3: EXAMPLE OF BLEU SCORE CALCULATION

Bi-gram sequence	Count	Clip count
the act	2	1
cat the	1	0
cat on	1	1
on the	1	1
the mat	1	1

As per the formula in Equation , Clip count is four and Count is six; therefore, the calculated BLEU score is 0.67. Similarly, the BLEU score for the text generation job in this research is calculated comparing the generated sentences with two of the most similar sentences as references. The result of the BLEU score for GCA, LaTextGAN and two SAGAN models are presented in Table 5.4.

Table 5.4: BLUE SCORE FOR DIFFERENT GAN MODELS

Model name	BLUE Score	
	Dataset 1	Dataset 2
CGA baseline model	0.63	0.41
SAGAN with CGA model	0.69	0.52
LaTextGAN baseline model	0.68	0.56
SAGAN with LaTextGAN model	0.70	0.62

In the results of all more models one important observation is the models without attention mechanism produce multiple of repetitive words in a sentence. While GAN models with attention module do not have this problem, and it is a significant improvement in GAN using attention module. Moreover, in all the four models, one common behavior is observed while generating the dialogues they generated the same sentences multiple times. So while choosing the results for evaluation only uniquely generated sentences are selected for further evaluation. Moreover, generating long sentences is also a very challenging task. All the outputs SAGAN using LaTextGAN model are having atleast six or more words, while models without attention unit can not produce sentences more than five words.

Visualization of Results

The training data and generated data is visualized using t-SNE visualizer. This visualization technique helps to visualize higher dimension data in low dimensions. This visualization helps us to understand how the training and generated data is distributed. This also helps us to understand whether the generated data is following the distribution of the training data properly or not. In Fig. 5.9 and 5.11 the distribution of training sentences and CAG,

LaTextGAN generated sentences are presented. In Fig. 5.10 and 5.12 the distribution of training sentences and SAGAN generated sentences are presented.

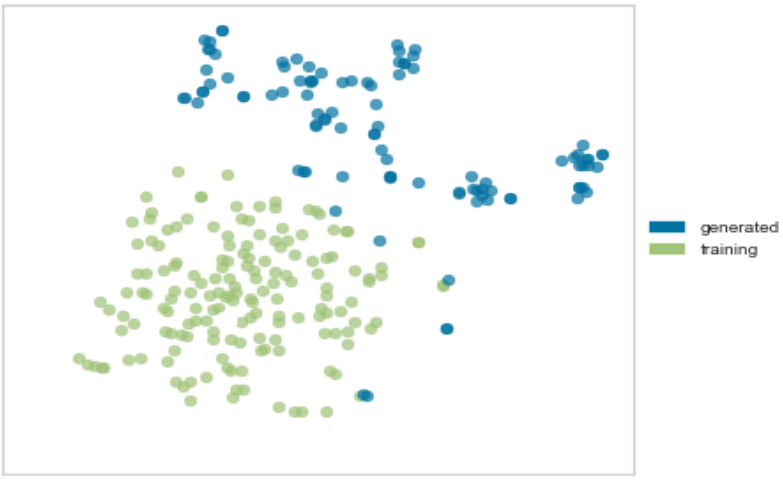


Figure 5.9: Distribution of training and result of GCA

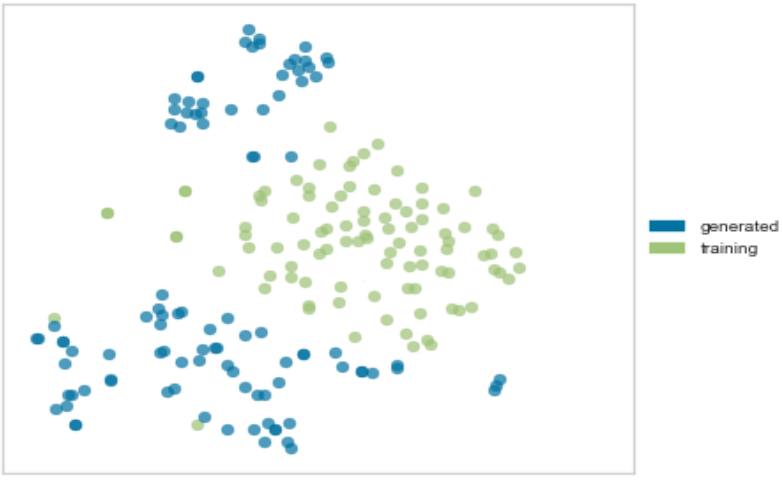


Figure 5.10: Distribution of training and result of SAGAN using GCA

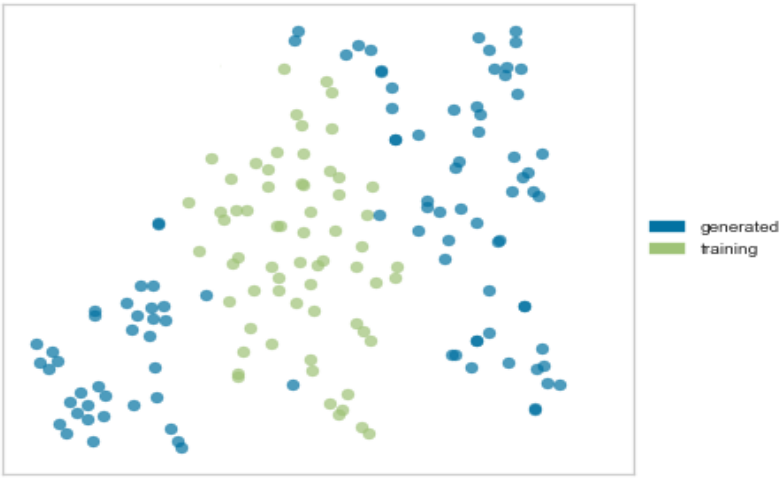


Figure 5.11: Distribution of training and result of LaTextGAN

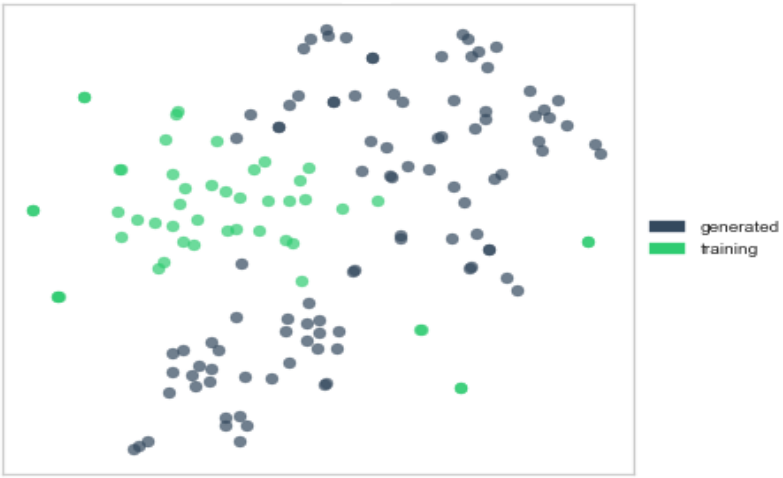


Figure 5.12: Distribution of training and result of SAGAN LaTextGAN

5.4 An Intrinsic Evaluation Metric

As discussed in the previous section, the BLEU score is the most common evaluation metric for text generation research. Other than BLEU, Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [94] score is also a very commonly used metric to evaluate

the quality of text summarization. ROUGE is very similar to the BLEU, and it is calculated as mentioned in Equation 5.21.

$$ROUGE\_n = \frac{\sum_{s \in \{RefSummaries\}} \sum_{n\_grams\, i \in S} min(count(i,X), count(i,S))}{\sum_{s \in \{RefSummaries\}} \sum_{n\_grams\, i \in S} count(i,S)} \tag{5.21}$$

In Equation 5.21 X is referring the summary of a document (D), generated by an algorithm whereas S is the summary of the same document (D) produced by an expert. The ROUGE score calculation procedure is similar to BLEU score, and in this case, also it is searching the same sequence in an *n_gram* sequence.

These methods are called intrinsic evaluation metrics of translation and summarization task. Although these are very commonly used, these methods are not developed to evaluate dialogues. Hence, there are certain aspects which these methods do not focus are essential for dialogue.

Lack of references: The performance of BLEU and ROUGE depends on the reference sentences. For randomly generated dialogues, it is very difficult to find out reference sentences to judge a newly generated sentence. So this is the primary reason for which BLEU and ROUGE are not appropriate metrics to evaluate dialogues generated by generative methods.

Informal nature of dialogue: Performance evaluation based on both BLEU and ROUGE heavily based on the order of the word sequences. While normal human conversations are often very informal and sometimes, the meaning of a sentence is clearly conveyed even though the sentence construction is not proper. In such a case, the BLEU and ROUGE score will be affected badly.

Semantic similarity: The commonly used evaluation metrics such as BLEU and ROUGE do not care about the semantic similarity of words. Suppose the reference sentence is "There is a cat on the mat," and the generated sentence is "The cat is on the mattress"; although the meaning is very similar as these evaluation metrics only care about exact words, they may consider it as a wrong word and may penalize it.

Length of dialogue: The length of a translated sentence or summary often plays an important role to evaluate the quality of the newly generated text. This is often evaluated by Bravity Penalty [114]. Bravity Penalty (BP) is calculated as per Equation 5.22.

$$BP = \begin{cases} 1, & \text{if } (output\_len > ref\_len) \\ \exp(1 - \frac{output\_len}{ref\_len}), & \text{otherwise} \end{cases} \tag{5.22}$$

$$BLEU\_BP = BP * BLEU \tag{5.23}$$

It is clear from the Equation 5.23 if the generated sentence is shorter than the reference sentence then BP is penalizing the generated sentence; on the other hand when the length of the generated sentence is longer than the reference sentence then there is no penalty on BLEU or ROUGE score.

Therefore by studying the previous research, it has been identified that there is a scope of further research to find out an evaluation metric for the dialogues generated by the generative methods. In this next part of this research article, an evaluation metric for any dialogues generation system is proposed. This metric is an intrinsic metric, and it is a combination of few already known metrics. The name of the proposed metric is Dialogue Evaluator (DE) metric.

5.4.1 Dialogue Evaluator

Dialogue Evaluator (DE) is the proposed metric to evaluate the dialogues generated by the generative model. It focuses on three critical characteristics of any dialogue, 1) sentence construction, 2) context, and 3) length.

Sentence construction: To evaluate the quality of a sentence, the primary requirement is the grammatical structure of the sentence should be proper. Although this does not mean we are strictly looking for a grammatically correct sentence, the sentence should be a minimum acceptable grammatical structure to become comprehensible. To find out the correctness of sentence structure, the generated sentences are parsed using NLTK and using Context Free Grammar (CFG) the validity of the sentence can be checked. It is called validity of the sentence.

Context score: The context of the generated sentence should be matched with the overall dialogue set. The context is measured using WordNet interface of NLTK. WordNet represents the semantic relations between words. At first the stop words are removed from the generated dialogue and then the average of the similarity score is calculated. The average similarity score is used as the context score for DE.

Length penalty: Unlike BLEU and ROUGE in case of DE, there is no fixed reference sentence, so the average length of the sentences in the training set is considered as the reference length of the dialogues. The length penalty (LP) parameter is very similar to BP. Length penalty is one of the lengths the same as the reference length of the sentence. Otherwise, the penalty is calculated as the exponential value inverse of the difference of length of reference and generated sentence.

$$LP = \begin{cases} 1, & \text{if } (output\_len = ref\_len) \\ \exp(\frac{1}{|ref\_len - output\_len|}), & \text{otherwise} \end{cases} \tag{5.24}$$

$$DE = validity * context\_socre * length\_penalty \tag{5.25}$$

As of the three parameters of equally important and independent so the DE score is calculated as in Equation 5.25. The DE score is calculated for the all the four GAN models have been discussed in this article. In Table 5.5 the DE score for each of the models are presented. It has been observed that the SAGAN models are having higher DE score than its counter parts; and LaTextGAN is having better DE score than GCA model.

Table 5.5: DE SCORE FOR DIFFERENT GAN MODELS

Model name	DE Score	
	Dataset 1	Dataset 2
CGA baseline model	0.63	0.41
SAGAN with CGA model	0.69	0.52
LaTextGAN baseline model	0.68	0.56
SAGAN with LaTextGAN model	0.70	0.62

5.5 Conclusion

Generation of the artificial dialogue text is a very important area of research in NLP. This research shows two new technique to generate dialogue for an intelligent conversation system using Self-Attention Generative Adversarial Network (SAGAN). Normally GAN is used to generate text which is very similar to human-generated text data. Addition of self-attention mechanism helps the GAN system to understand the small details of the dialogue system. Moreover, the context of a multi-turn dialogue system can be maintained by the usage of attention-mechanism. SAGAN is mainly used for computer vision problems. Using SAGAN for the generation of dialogue data is the most important contribution of this research work. Keeping in mind the nature of data to be generated by the generator and the evaluation function have been chosen. The generator is used for generating the discrete text

data while the evaluation function measures the quality of the generated data. Other than implementation of SAGAN this research also introduces a new intrinsic metric Dialogue Evaluator, which evaluates the quality of the dialogues generated by a generative model. This research has many directions to work in the future. Primarily, SAGAN can be used for different data generation techniques like inverse reinforcement learning and imitation learning. An extrinsic evaluation of function for dialogue evaluation can another important future work.

Chapter 6

Future Work and Conclusion

6.1 Future Work

So far, this article presented the research which has been carried out to study information and knowledge bots. In this study, the social media platform, Twitter has been used to study information Bots, whereas conversational agents are used as the knowledge bots. The entire study is based on the Data, Information, Knowledge, Wisdom (DIKW) model. This is a vast area of research, and here a small portion of each of the sections is discussed, and there are plenty of future research opportunity present for each of the areas. In this section of this article, the possible future research will be discussed.

6.1.1 Future Research on Information Bots

This research is concentrated only on Twitter as a social media platform. This research can be easily extended to other social media platforms such as Facebook. It will be interesting to study how information diffuses on Facebook and what factors are responsible for it. At the same time, it will also be an important point to investigate whether the parameters which are responsible for information diffusion on Twitter behaves the same way or not. If those parameters do not act in the same ways as they do on Twitter, then what is the reason behind it and how does the network structure of different social media responsible for information diffusion will also be an exciting subject to study. Other than extending all the research to another social media platform these research can be extended in new directions. The information bot research concentrated on two major subareas i) information diffusion and its different factors and, ii) characteristics analysis of social bots. The future research directions for each of these areas are discussed below.

While working on information diffusion, it has been observed that the diffusion pattern of information often depends on the nature of the information or news. A piece of sad news spreads much faster than news about tomorrow's weather. That is the reason we include

the sentiment of the tweets as an essential factor for information diffusion. Similarly, in the recent past, it has been observed that fake news on different social media platform became a widespread phenomenon. It also creates lots of confusion and misunderstanding among social media groups. The fake news is social media, also called the disinformation. Identifying fake news is an extremely challenging job, and currently, it is one of the most important areas of research in the social media research domain. There are really important tools present which help to identify the fake news such as: Grover [98] and GLTR [57]. There is a scope of improving these tools to identify fake news in the real-world. Cambridge University and Amazon released FEVER [131], which is the world's largest dataset for fact-checking. This dataset helps enormously to develop different machine learning models to identify fake news. In the coming years, it is expected to availability of more number of such datasets of fake news as the characteristics of fake news are changing very rapidly and gaining huge interest from the research community.

Along with machine learning or deep learning methods for fake news checking, researchers are also interested in using new technologies like Blockchain [41] to identify fake news. A very active research community has been developed who are working towards developing a distributed framework using Blockchain for identifying fake news in social media. Both these research directions can be used together to build a better platform to identify the fake news. Other than information diffusion, the second half of the information bot research discussed different characteristics of social bots. This is a new class of Twitter bot. It is challenging to identify them because they interact with other users as normal human behaves. In our research, we identified eight essential characteristics which are present in social bots. Our research is restricted to identifying the characteristics and finding the difference in pattern with traditional bots. A similar study can be performed to determine the difference between regular human users and social bots. This type of research will give a huge insight to identify the social bots because the differences in behavior between social bots and normal human users bots are not very clear. So it will be a hard task for the researchers to segregate social bots from normal Twitter users. The social bot research can also be extended to a classification job using machine learning and deep learning algorithms. The objective of the classifiers will be to classify different Twitter accounts into three classes such as: i) traditional bot, ii) social bot, and iii) normal users.

All these future research projects can be implemented for different commercial purposes. First of all, information diffusion is a very popular technique in business campaigning, product marketing and publicity events. Many e-commerce companies are already working in this direction and achieved a considerable amount of success. Second of all, identifying fake news in real-time is a real challenge in any social media or traditional media platform.

Moreover, there are lots of online, and some libraries, available to identify whether a Twitter account is a bot or a normal human user, but there is no application in my knowledge which can classify three types of Twitter users such as a traditional bot, social bot and not a bot. Finally, combining all the applications above there could be an end to end application where the fake news can be identified and at the same time, the community of bots spreading the fake news also can be identified.

6.1.2 Future Research on Knowledge Bots

The second part of this research article is concentrated on the development of knowledge bots. The success of a knowledge bot is hugely dependent on its knowledge base. To develop a knowledge bot or conversational agent, deep learning methods are becoming popular and also producing very good results. Training and testing of deep learning models need lots of data, and all conversation agents do not have enough data to train and test different deep learning methods. To solve this problem, two strategies are used such as i) transferring knowledge from a similar domain using transfer learning and ii) by generating synthetic data using the generative adversarial network (GAN).

The results of knowledge bot research show that attention mechanisms played an important role to generate unique and grammatically correct sentences. It has also been observed that there are plenty of scopes to improve the quality and diversity of the generated sentences using GANs. To enhance the performance of the current models, the use of Bidirectional Encoder Representations from Transformers (BERT) [44] model may play an important role. BRET model has helped the Machine Learning community to improve lots of state-of-the-art results in a wide variety of NLP tasks. Hence, while implementing SAGAN use of BRET will be an interesting and important result to observe. This may open new opportunities for future research.

The synthetic text data generation problem can be seen as data augmentation problems. Data augmentation is a prevalent and common technique in computer vision to increase the size and variety of training and testing dataset. The data augmentation in computer vision can be performed by some trivial operations on datasets such as rotating, changing the color scale, and displaying the partial image etc. This kind of change in text data makes no sense for most of the cases, and it may introduce noise or outliers. This is still an open problem for the NLP research community. There is some initial research that has been done towards the solution of text data augmentation problem. Data augmentation is a generic technique, which can be used for any NLP related task where data is not sufficient for training and testing. So there is a wide range of applications that will use this technique in the future.

For text data augmentation purposes [32], words of a sentence can be substituted by its synonyms. This is a very simple way to generate a similar kind of sentence. To enhance this technique using some important NLP support, word embedding techniques can be used with some threshold values. Similarly, WordNet databases also can be used to generate semantically similar sentences. Other than these techniques, a masked language model is important to generate synthetic data. Masked language transformer models such as BERT and ROBERTA [97] can be used to enhance and compare performance with different state-of-the-art models. Back translation and text surface transformation [32] are two important techniques to augment text data. These two techniques focus on ambiguity in verbal form expansion. Random noise injection is also a method to generate augmented text data. In this method intentionally some error and noise is added in the text data. Data augmentation can also be performed by manipulating the syntax tree of a sentence.

Other than SAGAN models in the discussion of knowledge bots a new metric named Dialogue Evaluator (DE) is also introduced. The objective of this metric is to evaluate the synthetically generated dialogues. This evaluation metric is very specific to this task. As there are few downsides observed in BLEU or ROUGE (popularly used evaluation metrics), the introduction of DE is an important part of this research. DE considers three important factors of the dialogues generated by GANs such as: i) the validity of a sentence, ii) context of the generated words, and iii) length of the sentences. These scores are called intrinsic metrics of the text data of the generated dialogues.

There are lots of scope of improvement in the definition of DE. First of all, DE used length penalty as the parameter to determine whether the length of the generated dialogues match with the desired length or not. For this calculation desired length is always considered as the average length of the entire dataset. The length penalty parameter can be improved and a generic length penalty function can be used. Other than that, the computation time of DE is very high, as the calculation time for each of the parameters is high. In normal practice, extrinsic metrics are considered as easier to calculate and more generic in nature. Hence, a generic extrinsic metric for dialogue evaluation is important future work.

6.2 Conclusion

The study of information bots and knowledge bots is an attempt to explore the Data Information Knowledge Wisdom model for massive data-intensive platforms. Social media network Twitter is used for the significant part of this research as the source of data and the information related to the data. The data is collected using the APIs provided by Twitter and the hashtags in the tweets are considered as the subject or context of the data. So data collected through the Twitter APIs became a piece of information when a relevant hashtag is identified with it. In the course of the research, it has been observed that some of the Twitter accounts act as a conversational agent and that observations give the idea of the next direction of this research. The next half of the research is concentrated on analyzing the information and learning from it. A conversational agent is considered as the representative of knowledge bots in this research. This research concentrated on improving the knowledgebase of the conversational agents or in other words chatbots. To improve the quality and quantity of the knowledgebase of conversational agents, two strategies are implemented.

The study of information bots starts with identifying different characteristics and patterns of information diffusion. It also investigates the various factors that govern these characteristics and patterns. In this study, it has been identified that three primary factors are responsible for information diffusion on Twitter. These three factors are: the sentiment of tweets, the influence of a user and volume of the tweet about a certain subject. Each of these factors consists of multiple parameters such as: the sentiment of tweets with five parameters (positive percentage, neutral percentage, negative percentage, positive average score, neutral average score, negative average score); the influence of a user who has two parameters (direct influence user, indirect influence user); and volume of tweets on a subject has two parameters (number of tweets, number of retweets). To predict the pattern of the information diffusion with respect to each of these factors, the traditional time series method is used initially as the baseline method to compare results using deep learning methods. Sentiment analysis of tweets is an important part of this research as there are six parameters responsible for determining the influence of sentiments in information diffusion. A new method of sentiment analysis of tweets is introduced and compared with the traditional methods. For the implementation of a time series prediction ARIMA model is used, and LSTM is used as the deep learning method. The patterns of information diffusion are identified using the Dynamic Time Wrapping clustering method. To find the optimum number of the clusters, six cluster validity indices or CVIs are used. The results show that for each of the factors, the prediction of LSTM is better than the traditional ARIMA model.

The further study of the information bots focuses on the network structure, information diffusion pattern and content of the information of social bots. This is a new type of intelligent Twitter bot which imitates human behavior on Twitter and they are different from traditional Twitter bots which are responsible for only spamming. To understand the difference of characteristics of traditional bots and social bots in network structure, information diffusion pattern and content of the information there are eight research questions that are discussed. The analysis of the evolved network structure of social bots has been done by answering three important questions: i) How does the new wave of social bots differ from traditional bots in terms of social network statistics, their organization of Core-Periphery structure? ii) How embedded are the social bots in their social as well as communication networks? iii) How do the networks of the social bots perform under Robustness attack? The study of the information diffusion and communication patterns of the social bots is done by answering the following questions: iv) What does the information diffusion patterns of the social bots look like? v) Do the bots have different communication leaders across different forms of communication networks? vi) How homogenous and distributed are the categories of tweets coming from bots, compared to their traditional counterparts? A detailed content analysis of the tweets produced by those bots has been done by answering the questions: vii) Do the social bots have any specific patterns of topic distribution over time? viii) Do the bots have some community-specific content spreading behavior? To explore each of the characteristics there are five networks created using the bots such as: i) Social network, ii) Retweet Network, iii) Mention Network, iv) URL Network, and v) Hashtag Network. Each network is evaluated by appropriate metrics such as core-periphery interaction, K-core decomposition, robustness attack test and information diffusion timeline. To generate different social networks between the bots, graph slicing techniques are used and NetworkX library is used for the implantation of these algorithms. The experiment is carried on two classes of Twitter bots such as political bots and advertisement bots. Social bots who are involved in a political campaign are compared with traditional bots who are involved in such activities. Similarly, the characteristics of social and traditional bots who are involved in advertisements are also compared. The data is collected from the previous research where the traditional and social bots are already identified. Although all the bots mentioned in the previous research paper are not currently available as Twitter has its own bot detection system and that removes a huge number of accounts from Twitter every year. Converting the experimental findings of this study to quantitative and statistical measures, which could possibly be extended to a real-time expert detection system of social bots, is a major remaining challenge as we look forward to joining forces on bringing down these new waves of bots on Twitter.

The second half of the research is concentrated on knowledge bots. The conversational agent or the chatbots are considered as the knowledge bots here and the discussion is restricted only with the text data. The central objective of this research is to enhance the quality and size of the knowledge base so that it will be suitable to use for deep learning models. The first half of the research deals with Goal-Oriented (GO) conversational system or GO chatbot. This research shows how to handle the inadequate data problem using transfer learning and attention mechanism. Application of transfer learning allows the GO chatbot to transfer the common knowledge of one domain to another domain, which solves the problem of inadequate data for a particular domain. On the other hand, the attention mechanism helps the model to perform domain-specific chatting. The proposed model produces a better result on datasets which are used in the previous research work and also for a newly introduced dataset for organ transplant information. Two main contributions of this research are, using transfer learning and attention mechanism for GO chatbot, and introducing a new dataset for organ transplant information.

By using GANs and SAGANs synthetic data is generated to solve the inadequate data problem in the second part of the knowledge bots study. GANs are popularly used to generate image data but not very popular for generating text data. In this research work, we have used GANs to generate text data. In this context, the text data which are generated by the GANs and SAGANs are the inputs to develop a knowledgebase. Eventually, these generated text data will be the dialogues of the conversational agents. So one of the important characteristics of the generated text is shorter in size. This nature of the text made the problem more difficult. To keep the size of the generated dialogue and context as per the training dialogue set, SAGAN is used. SAGAN is very popular for image generation and it focuses on the small details of images. The SAGAN is used here to keep the small details of the training dataset in the generated text data. In this research, two SAGAN models are implemented based on GCA and LaTextGAN. In both the models, the attention layer is added to make it a SAGAN model. Two different datasets are used for all the experiments. All the four models (GCA, LaTextGAN, SAGAN with GCA, SAGAN with LaTextGAN) are trained and tested using these two datasets. The first question and answer dataset is published with GCA and the GCA model is used as a baseline model. Ubuntu Dialogue Corpus is used as the second dataset, it is a publicly available dataset. While comparing the results of all the four models, SAGAN with LaTextGAN model's performance is better than the rest of the GANs and SAGAN models. Overall, the performance of SAGANs is better than GANs. To compare the results of all the four models BLEU and ROUGE score have been used. As these evaluator metrics are suitable for NMT and text summarization work, there are some limitations identified to evaluate a dialogue generation system. The BLUE

and ROUGE have limitations because the dialogues do not have any reference to compare, sometimes dialogues are very informal, and the length of the dialogues often vary. To solve these problems a new evaluation metric is introduced named Dialogue Evaluator (DE). This is an intrinsic metric of the dialogues generated by the GANs and SAGANs. The results of GANs and SAGANs in this experiment are also compared and present the SAGANs are producing better results than GANs. Although the final results for BLEU, ROUGE and DE are the same, DE gives a better insight into the quality of the dialogues. The use of SAGAN to generate text data and the proposed evaluation metric (DE) are the two most important contributions of knowledge bots study.

To analyze and implement these various methodologies, different machine learning, deep learning and reinforcement learning techniques are used, and encouraging experimental results are presented that demonstrate the great potential of our approaches in applications using information and knowledge bots.

BIBLIOGRAPHY

[1] Ai adoption advances, but foundational barriers remain. https://www.mckinsey.com/featured-insights/artificial-intelligence/ai-adoption-advances-but-foundational-barriers-remain/. 2018.

[2] Amartya hatua. transplant candidate registration worksheet information for five different organs such as kidney, lung, heart, pancreas and liver. https://github.com/amartyahatua/chatbot_transfer_attention/. 2019.

[3] Botcheck.me. https://www.rand.org/research/projects/truth-decay/fighting-disinformation/search/items/botcheckme.html. 2017.

[4] Botometer. https://botometer.iuni.iu.edu/#!/. 2018.

[5] Center for disease control and prevention. https://www.cdc.gov/%20coronavirus/2019-ncov/index.html,%202020./. 2020.

[6] Google cloud. https://cloud.google.com/apis/. 2020.

[7] How much data is generated every minute? https://www.socialmediatoday.com/news/how-much-data-is-generated-every-minute-infographic-1/525692. 2018.

[8] Influence tracker. http://influencetracker.com/. 2017.

[9] Information diffusion. https://github.com/amartyahatua/informationdiffusion/. 2018.

[10] Mind matters. if a robot read the news, would you notice a difference? https://www.nytimes.com/2019/02/05/business/media/artificial-intelligence-journalism-robots.html/. 2019.

[11] The rise of the robot reporter. https://mindmatters.ai/2018/11/if-a-robot-read-the-news-would-you-notice-a-difference/. 2018.

[12] Transaction or knowledge chatbot? moving beyond the siri syndrome. https://www.telerik.com/blogs/transactional-or-knowledge-chatbot-moving-beyond-siri-syndrome/. 2018.

[13] Tweetbotornot. https://github.com/mkearney/tweetbotornot. 2018.

[14] Twitter. https://twitter.com/home/. 2006.

[15] Twitter docs. https://developer.twitter.com/en/docs/. 2006.

[16] Woebot. https://woebot.io/. 2020.

[17] Amit A Amleshwaram, AL Narasimha Reddy, Sandeep Yadav, Guofei Gu, and Chao Yang. Cats: Characterizing automation of twitter spammers. In *COMSNETS*, pages 1–10, 2013.

[18] Olatz Arbelaitz, Ibai Gurrutxaga, Javier Muguerza, JesúS M PéRez, and IñIgo Perona. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256, 2013.

[19] Dimitrios Asteriou and Stephen G Hall. *Applied econometrics*. Macmillan International Higher Education, 2015.

[20] Lars Backstrom, Dan Huttenlocher, Jon Kleinberg, and Xiangyang Lan. Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM*

SIGKDD international conference on Knowledge discovery and data mining, pages 44–54, 2006.

[21] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[22] Gene Bellinger, Durval Castro, and Anthony Mills. Data, information, knowledge, and wisdom, 2004.

[23] Ahmida Bendjoudi. Einstein's framework for natural language processing, 2020.

[24] Alessandro Bessi and Emilio Ferrara. Social bots distort the 2016 us presidential election online discussion. *First Monday*, 21(11-7), 2016.

[25] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[26] Mikael Boden. A guide to recurrent neural networks and backpropagation. *the Dallas project*, 2002.

[27] Yazan Boshmaf, Ildar Muslukhov, Konstantin Beznosov, and Matei Ripeanu. The socialbot network: when bots socialize for fame and money. In *Proceedings of the 27th annual computer security applications conference*, pages 93–102, 2011.

[28] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[29] Gwilym M. Jenkins Gregory C. Reinsel Box, George EP and Greta M. Ljung. Time series analysis: forecasting and control. In *John Wiley & Sons.*, 2015.

[30] Danah Boyd, Scott Golder, and Gilad Lotan. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *2010 43rd Hawaii international conference on system sciences*, pages 1–10. IEEE, 2010.

[31] Eoin Brophy, Zhengwei Wang, and Tomas E Ward. Quick and easy time series generation with established image-based gans. *arXiv preprint arXiv:1902.05624*, 2019.

[32] Amit Chaudhary. A visual survey of data augmentation in nlp, 2020. `https://amitness.com/2020/05/data-augmentation-for-nlp`.

[33] Liqun Chen, Shuyang Dai, Chenyang Tao, Haichao Zhang, Zhe Gan, Dinghan Shen, Yizhe Zhang, Guoyin Wang, Ruiyi Zhang, and Lawrence Carin. Adversarial text generation via feature-mover's distance. In *Advances in Neural Information Processing Systems*, pages 4666–4677, 2018.

[34] Zeyuan Chen, Shaoliang Nie, Tianfu Wu, and Christopher G Healey. High resolution face completion with multiple controllable attributes via fully end-to-end progressive generative adversarial networks. *arXiv preprint arXiv:1801.07632*, 2018.

[35] Junsuk Choe, Song Park, Kyungmin Kim, Joo Hyun Park, Dongseob Kim, and Hyunjung Shim. Face generation for low-shot learning using generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1940–1948, 2017.

[36] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018.

[37] François Chollet et al. Keras: Deep learning library for theano and tensorflow. *URL: https://keras. io/k*, 7(8):T1, 2015.

[38] Richard Colbaugh and Kristin Glass. Early warning analysis for social diffusion events. *Security Informatics*, 1(1):18, 2012.

[39] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. Dna-inspired online behavioral modeling and its application to spambot detection. *IEEE Intelligent Systems*, 31(5):58–64, 2016.

[40] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race. In *Proceedings of the 26th international conference on world wide web companion*, pages 963–972, 2017.

[41] Michael Crosby, Pradan Pattanayak, Sanjeev Verma, Vignesh Kalyanaraman, et al. Blockchain technology: Beyond bitcoin. *Applied Innovation*, 2(6-10):71, 2016.

[42] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Russ R Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *Advances in neural information processing systems*, pages 6510–6520, 2017.

[43] Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. Generalized louvain method for community detection in large networks. In *2011 11th international conference on intelligent systems design and applications*, pages 88–93. IEEE, 2011.

[44] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[45] David M Dickinson, Mary D Ellison, and Randall L Webb. Data sources and structure. 2003.

[46] Brian Dolhansky and Cristian Canton Ferrer. Eye in-painting with exemplar generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7902–7911, 2018.

[47] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. *arXiv preprint arXiv:1802.04208*, 2018.

[48] Hao Dong, Simiao Yu, Chao Wu, and Yike Guo. Semantic image synthesis via adversarial learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5706–5714, 2017.

[49] Andrej Duh, Marjan Slak Rupnik, and Dean Korošak. Collective behavior of social bots is encoded in their temporal twitter activity. *Big data*, 6(2):113–123, 2018.

[50] Rob A Dunne and Norm A Campbell. On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function. In *Proc. 8th Aust. Conf. on the Neural Networks, Melbourne*, volume 181, page 185. Citeseer, 1997.

[51] Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*, 2015.

[52] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[53] William Fedus, Ian Goodfellow, and Andrew M Dai. Maskgan: Better text generation via filling in the_. *arXiv preprint arXiv:1801.07736*, 2018.

[54] Emilio Ferrara and Zeyao Yang. Quantifying the effect of sentiment on information diffusion in social media. *PeerJ Computer Science*, 1:e26, 2015.

[55] Michelle Forelle, Phil Howard, Andrés Monroy-Hernández, and Saiph Savage. Political bots and the manipulation of public opinion in venezuela. *arXiv preprint arXiv:1507.07109*, 2015.

[56] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[57] Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*, 2019.

[58] Saptarshi Ghosh, Bimal Viswanath, Farshad Kooti, Naveen Kumar Sharma, Gautam Korlam, Fabricio Benevenuto, Niloy Ganguly, and Krishna Phani Gummadi. Understanding and combating link farming in the twitter social network. In *Proceedings of the 21st international conference on World Wide Web*, pages 61–70, 2012.

[59] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[60] Mark Granovetter. Threshold models of collective behavior. *American journal of sociology*, 83(6):1420–1443, 1978.

[61] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

[62] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[63] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[64] Md Akmal Haidar and Mehdi Rezagholizadeh. Textkd-gan: Text generation using knowledge distillation and generative adversarial networks. In *Canadian Conference on Artificial Intelligence*, pages 107–118. Springer, 2019.

[65] Giannis Haralabopoulos and Ioannis Anagnostopoulos. On the information diffusion between web-based social networks. In *International Conference on Web Information Systems Engineering.*, pages 14–26. Springer, Cham, 2014.

[66] Ioannis Anagnostopoulos Haralabopoulos, Giannis and Sherali Zeadally. Lifespan and propagation of information in on-line social networks: A case study based on reddit. In *Journal of network and computer applications.*, pages 88–100, 2015.

[67] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

[68] Kay Gregor Hartmann, Robin Tibor Schirrmeister, and Tonio Ball. Eeg-gan: Generative adversarial networks for electroencephalograhic (eeg) brain signals. *arXiv preprint arXiv:1806.01875*, 2018.

[69] Amartya Hatua, Trung T Nguyen, and Andrew H Sung. Information diffusion on twitter: Pattern recognition and prediction of volume, sentiment, and influence. In *Proceedings of the Fourth IEEE/ACM International Conference on Big Data Computing, Applications and Technologies*, pages 157–167, 2017.

[70] Amartya Hatua, Trung T Nguyen, and Andrew H Sung. Dialogue generation using self-attention generative adversarial network. In *2019 IEEE International Conference on Conversational Data & Knowledge Engineering (CDKE)*, pages 33–38. IEEE, 2019.

[71] Amartya Hatua, Trung T Nguyen, and Andrew H Sung. Goal-oriented conversational system using transfer learning and attention mechanism. In *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0099–0104. IEEE, 2019.

[72] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[73] Kai Hu, Zhenzhen Zhang, Xiaorui Niu, Yuan Zhang, Chunhong Cao, Fen Xiao, and Xieping Gao. Retinal vessel segmentation of color fundus images using multiscale convolutional neural network with an improved cross-entropy loss function. *Neurocomputing*, 309:179–191, 2018.

[74] Vladimir Ilievski, Claudiu Musat, Andreea Hossmann, and Michael Baeriswyl. Goal-oriented chatbot dialog management bootstrapping with transfer learning. *arXiv preprint arXiv:1802.00500*, 2018.

[75] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[76] Swami Iyer, Timothy Killingback, Bala Sundaram, and Zhen Wang. Attack robustness and centrality of complex networks. *PloS one*, 8(4):e59613, 2013.

[77] Nikolay Jetchev, Urs Bergmann, and Roland Vollgraf. Texture synthesis with spatial generative adversarial networks. *arXiv preprint arXiv:1611.08207*, 2016.

[78] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. Inferring lockstep behavior from connectivity pattern in large graphs. *Knowledge and Information Systems*, 48(2):399–428, 2016.

[79] Felix Juefei-Xu, Rahul Dey, Vishnu Naresh Boddeti, and Marios Savvides. Rankgan: a maximum margin ranking gan for generating faces. In *Asian Conference on Computer Vision*, pages 3–18. Springer, 2018.

[80] Eleanna Kafeza, Andreas Kanavos, Christos Makris, and Pantelis Vikatos. Predicting information diffusion patterns in twitter. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 79–89. Springer, 2014.

[81] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[82] Tuja Khaund, Kiran Kumar Bandeli, Muhammad Nihal Hussain, Adewale Obadimu, Samer Al-Khateeb, and Nitin Agarwal. Analyzing social and communication network structures of social bots and humans. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 794–797. IEEE, 2018.

[83] Mohammad Ahangar Kiasari, Dennis Singh Moirangthem, and Minho Lee. Coupled generative adversarial stacked auto-encoder: Cogasa. *Neural Networks*, 100:1–9, 2018.

[84] Minho Kim and RS Ramakrishna. New indices for cluster validity assessment. *Pattern Recognition Letters*, 26(15):2353–2363, 2005.

[85] Matt J Kusner and José Miguel Hernández-Lobato. Gans for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*, 2016.

[86] Christoph Lassner, Gerard Pons-Moll, and Peter V Gehler. A generative model of people in clothing. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 853–862, 2017.

[87] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[88] Kyumin Lee, Brian David Eoff, and James Caverlee. Seven months with the devils: A long-term study of content polluters on twitter. In *Fifth international AAAI conference on weblogs and social media*, 2011.

[89] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *International Conference on Artificial Neural Networks*, pages 703–716. Springer, 2019.

[90] Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. End-to-end task-completion neural dialogue systems. *arXiv preprint arXiv:1703.01008*, 2017.

[91] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. Generative face completion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3911–3919, 2017.

[92] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727, 2015.

[93] Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017.

[94] Chin-Yew Lin, Guihong Cao, Jianfeng Gao, and Jian-Yun Nie. An information-theoretic approach to automatic evaluation of summaries. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 463–470, 2006.

[95] Po-Ching Lin and Po-Min Huang. A study of effective features for detecting long-surviving twitter spam accounts. In *2013 15th International Conference on Advanced Communications Technology (ICACT)*, pages 841–846. IEEE, 2013.

[96] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in neural information processing systems*, pages 700–708, 2017.

[97] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[98] Gui-Lu Long. Grover algorithm with zero theoretical failure rate. *Physical Review A*, 64(2):022307, 2001.

[99] Pauline Luc, Camille Couprie, Soumith Chintala, and Jakob Verbeek. Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408*, 2016.

[100] Oswaldo Ludwig. End-to-end adversarial learning for generative conversational agents. *arXiv preprint arXiv:1711.10122*, 2017.

[101] Liqian Ma, Xu Jia, Qianru Sun, Bernt Schiele, Tinne Tuytelaars, and Luc Van Gool. Pose guided person image generation. In *Advances in neural information processing systems*, pages 406–416, 2017.

[102] Shuang Ma, Jianlong Fu, Chang Wen Chen, and Tao Mei. Da-gan: Instance-level image translation by deep attention generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5657–5666, 2018.

[103] Dwarikanath Mahapatra, Behzad Bozorgtabar, and Rahil Garnavi. Image super-resolution using progressive generative adversarial networks for medical image analysis. *Computerized Medical Imaging and Graphics*, 71:30–39, 2019.

[104] Dwarikanath Mahapatra, Behzad Bozorgtabar, Sajini Hewavitharanage, and Rahil Garnavi. Image super resolution using generative adversarial networks and local saliency maps for retinal image analysis. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 382–390. Springer, 2017.

[105] Marina Meilă. Comparing clusterings by the variation of information. In *Learning theory and kernel machines*, pages 173–187. Springer, 2003.

[106] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[107] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[108] Fred Morstatter, Liang Wu, Tahora H Nazer, Kathleen M Carley, and Huan Liu. A new approach to bot detection: striking the balance between precision and recall. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 533–540. IEEE, 2016.

[109] Ramesh Nallapati, Bing Xiang, and Bowen Zhou. Sequence-to-sequence rnns for text summarization. 2016.

[110] Nasir Naveed, Thomas Gottron, Jérôme Kunegis, and Arifah Che Alhadi. Bad news travel fast: A content-based analysis of interestingness on twitter. In *Proceedings of the 3rd international web science conference*, pages 1–7, 2011.

[111] Trung T Nguyen, Amartya Hatua, Asheshbabu Pothuraju, and Andrew H Sung. Influence modeling, volume prediction and sentiment analysis of short texts on twitter.

[112] Vu Dung Nguyen, Blesson Varghese, and Adam Barker. The royal birth of 2013: Analysing and visualising public sentiment in the uk using twitter. In *2013 IEEE International Conference on Big Data*, pages 46–54. IEEE, 2013.

[113] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651, 2017.

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

[115] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[116] Pujan Paudel, Trung T Nguyen, Amartya Hatua, and Andrew H Sung. How the tables have turned: studying the new wave of social bots on twitter using complex network analysis

techniques. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 501–508, 2019.

[117] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[118] Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. *arXiv preprint arXiv:1704.03084*, 2017.

[119] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[120] François Petitjean, Alain Ketterlin, and Pierre Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, 44(3):678–693, 2011.

[121] Julio Cesar Louzada Pinto and Tijani Chahed. Modeling multi-topic information diffusion in social networks using latent dirichlet allocation and hawkes processes. In *2014 Tenth International Conference on Signal-Image Technology and Internet-Based Systems*, pages 339–346. IEEE, 2014.

[122] Ben Poole, Alexander A Alemi, Jascha Sohl-Dickstein, and Anelia Angelova. Improved generator objectives for gans. *arXiv preprint arXiv:1612.02780*, 2016.

[123] Zhaofan Qiu, Yingwei Pan, Ting Yao, and Tao Mei. Deep semantic hashing with generative adversarial networks. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 225–234, 2017.

[124] Jacob Ratkiewicz, Michael D Conover, Mark Meiss, Bruno Gonçalves, Alessandro Flammini, and Filippo Menczer Menczer. Detecting and tracking political abuse in social media. In *Fifth international AAAI conference on weblogs and social media*, 2011.

[125] Ray Reagans and Bill McEvily. Network structure and knowledge transfer: The effects of cohesion and range. *Administrative science quarterly*, 48(2):240–267, 2003.

[126] Radim Rehurek and Petr Sojka. Gensim statistical semantics in python. *Retrieved from genism.org*, 2011.

[127] Jennifer E Rowley and Richard J Hartley. *Organizing knowledge: an introduction to managing access to information*. Ashgate Publishing, Ltd., 2008.

[128] Sandro Saitta, Benny Raphael, and Ian FC Smith. A bounded index for cluster validity. In *International workshop on machine learning and data mining in pattern recognition*, pages 174–187. Springer, 2007.

[129] Alexis Sardá-Espinosa. Comparing time-series clustering algorithms in r using the dtwclust package. *R package vignette*, 12:41, 2017.

[130] Jost Schatzmann, Blaise Thomson, and Steve Young. Error simulation for training statistical dialogue systems. In *2007 IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU)*, pages 526–531. IEEE, 2007.

[131] Tal Schuster, Darsh J Shah, Yun Jie Serene Yeo, Daniel Filizzola, Enrico Santus, and Regina Barzilay. Towards debiasing fact verification models. *arXiv preprint arXiv:1908.05267*, 2019.

[132] Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[133] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[134] Nasim Souly, Concetto Spampinato, and Mubarak Shah. Semi supervised semantic segmentation using generative adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5688–5696, 2017.

[135] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.

[136] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[137] Ange Tato and Roger Nkambou. Improving adam optimizer. 2018.

[138] Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. Sentiment strength detection in short informal text. *Journal of the American society for information science and technology*, 61(12):2544–2558, 2010.

[139] Matteo Tomei, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Art2real: Unfolding the reality of artworks via semantically-aware image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5849–5859, 2019.

[140] Dustin Tran, Rajesh Ranganath, and David Blei. Hierarchical implicit models and likelihood-free variational inference. In *Advances in Neural Information Processing Systems*, pages 5523–5533, 2017.

[141] Gokhan Tur and Renato De Mori. *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons, 2011.

[142] Trupti V Udapure, Ravindra D Kale, and Rajesh C Dharmik. Study of web crawler and its different types. *IOSR Journal of Computer Engineering (IOSR-JCE)*, 16(1):01–05, 2014.

[143] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

[144] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[145] Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

[146] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances in neural information processing systems*, pages 613–621, 2016.

[147] Mike Wallace. Jawbone java wordnet api, 2007.

[148] Jing Wang and Ioannis Ch Paschalidis. Botnet detection based on anomaly and community detection. *IEEE Transactions on Control of Network Systems*, 4(2):392–404, 2016.

[149] Nan Wang, Blesson Varghese, and Peter D Donnelly. A machine learning analysis of twitter sentiment to the sandy hook shootings. In *2016 IEEE 12th International Conference on e-Science (e-Science)*, pages 303–312. IEEE, 2016.

[150] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018.

[151] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[152] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745*, 2015.

[153] Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*, 2016.

[154] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6721–6729, 2017.

[155] Jaewon Yang and Jure Leskovec. Modeling information diffusion in implicit networks. In *2010 IEEE International Conference on Data Mining*, pages 599–608. IEEE, 2010.

[156] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5485–5493, 2017.

[157] Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

[158] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[159] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[160] Rose Yu, Xinran He, and Yan Liu. Glad: group anomaly detection in social media analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 10(2):1–22, 2015.

[161] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International Conference on Machine Learning*, pages 7354–7363, 2019.

[162] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 5907–5915, 2017.

[163] Yizhe Zhang, Zhe Gan, and Lawrence Carin. Generating text via adversarial training. In *NIPS workshop on Adversarial Training*, volume 21, 2016.

[164] Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation. *arXiv preprint arXiv:1706.03850*, 2017.

[165] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, 2016.

[166] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pages 597–613. Springer, 2016.

[167] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[168] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in neural information processing systems*, pages 465–476, 2017.

[169] Dmitry Zinoviev. *Complex network analysis in Python: Recognize-construct-visualize-analyze-interpret*. Pragmatic Bookshelf, 2018.