UNIVERSITY OF CYPRUS

Fake News Detection - Social Network Perspective

by

Rafael Andreou

A thesis submitted in partial fulfillment for the degree of Computer Science

in the

Faculty Of Pure And Applied Sciences
Computer Science Department

May 2019

Declaration of Authorship

I, RAFAEL ANDREOU , declare that this thesis titled, 'THESIS TITLE' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

“You can have data without information, but you cannot have information without data.”

Napoleon Bonaparte, French military and political leader

Abstract

Undoubtedly nowadays misinformation and falsity have invaded our lives. People have their minds daily affected and influenced by information that desires to confuse them about the truth on serious events, especially political and economic activities. That phenomenon has motivated researchers to study and analyze the problem and try to create an automated tool that will be able to detect fake-news disseminators and so tackle-reduce the spread of falsity over social networks. Despite the many tries and approaches taken by a lot, due to the problem's complexity falsity travels day by day at rapid acceleration. In our research, we focus on the Social Network perspective of Fake News on Twitter and try to determine those features that differentiate fake news disseminators from trusted accounts using a different approach from the literature. We construct our own Dataset after continuously crawling Twitter for a month. We then label the collected tweets according to whether the tweet contains URLs from untrusted-suspicious domains. Following up, we split into sliding windows our tweets and run a diffusion process on each window using De Groot's Model which introduces a simple mechanism of opinion propagation. We end up with a blacklist of the top fake news disseminator accounts. Furthermore, we split them into 4 buckets using dissemination probability intervals of 0.25. Finally, we take the top 1000 active accounts of each bucket and query their recent timeline history and export specific features which are then plotted to analyze their difference from trusted accounts.

Acknowledgements

I would like to express my deep gratitude to Professor Mr.Georgios Pallis, my research supervisor, for providing me an opportunity to be a part of this wonderful project and such a trend topic nowadays, Fake News. I am highly indebted to his guidance and support in getting familiar with a topic I did not have any knowledge or experience before.

My special thanks also go to my co-operator and friend Mr.Demetris Paschalides. His encouragement, instructions and suggestions were vital in the completion of this project and being far more experienced than me, he willingly helped me out with his abilities and daily support.

Lastly, my parents George and Georgia, receive my deepest gratitude and love for their dedication and the many years of support during my undergraduate studies that provided the foundation for this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

1.1 Motivation

Traditional media consists of mostly nameless and faceless people deciding what does and does not get printed and broadcasted. In this new age of the internet and a variety of social media, creation, and consumption of news and information in our society is evolving. The accelerated transformation of old print media toward online portals is a trend nowadays. For instance, 62 percent of U.S. adults watch news on social media in 2016, while in 2012, only 49 percent reported seeing news on social media. In addition, it is found that social media now beats television as the leader news source. Social networks are now capable of providing us with the ability to stay up to date with the latest news and information that is spreading rapidly all over the world. People may use them not just to be informed but also to broadcast and share breaking news, to describe an event or event to express their opinion and feelings on serious matters.

However, due to the fact that it is cheap to provide news online and much faster and easier to disseminate through social media, despite their benefits, social networkWAs can also be used by people who want to take advantage of their popularity in a bad way to affect and influence people's opinions by spreading falsity or misinformation. Twitter and Facebook are the most trending social networks and are being used by billions. Thus, they can also be seen as a fertile breeding ground for spreading false and fake news, reaching a huge number of people in a very short span of time.

A small definition of fake news is news articles that are purposely and verifiably false and could mislead readers [1].Disseminating falsity over those social networks is believed to have serious consequences and impacts on individuals and societies, especially in times of disasters or events that involve national security, or other popular political and economic activities. Normal people who live in this rapidly-growing online word can have their mind, opinions or choices altered over serious situations across the world, breaking in that way the authenticity balance of the news ecosystem. For instance, it is indisputable that the most popular fake news was even more widely spread on Facebook than the most popular trustworthy mainstream news during the U.S. 2016 presidential election. Figure - 1.1 show how much influence bogus news stories had in the final months of the elections campaings. Fake news is also changing the way people evaluate and react to real news. For example, some fake news' intention is to trigger people's distrust and make them confused, hindering their abilities to distinguish what is true from what is not. Our economies are not immune to the spread of fake news either, with fake news being connected to stock market fluctuations and massive trades.

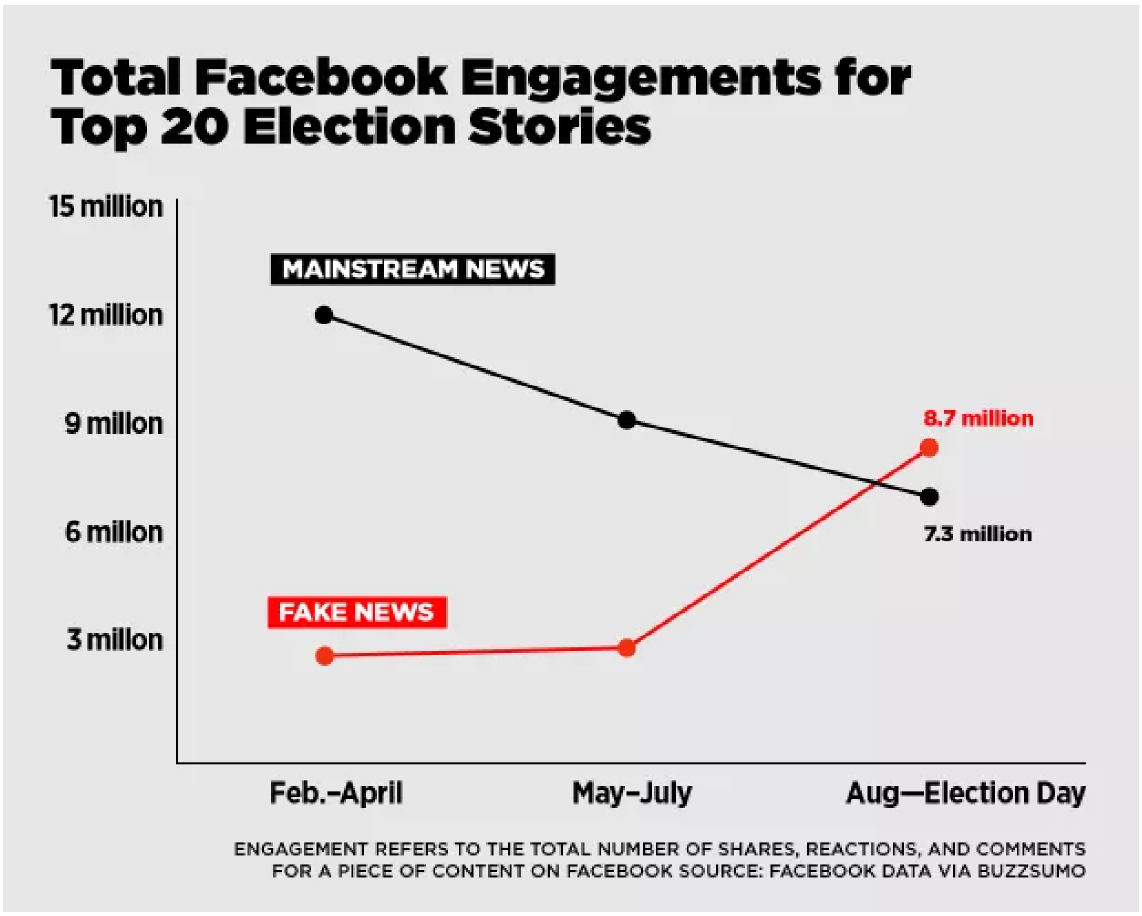


FIGURE 1.1: Top 20 fake news stories outperformed real news at the end of the 2016 campaign

All the above bad effects of misinformation and fake news lead to the need of fake news detection on social networks-media which has recently become emerging research that is attracting great attention. Consequently, taking into consideration all of its bad effects in our individual lives and societies, a high-demand on detecting fake-news disseminators and so tackle-reduce the spread of falsity over those social networks.

1.2 Challenges

All researches related to fake news detection or fake news tackling have many challenges. First and foremost a continuous and valid Dataset is required and hard to find. It is very important in such research for the Dataset to be continued, meaning having collected a stream of data (in our case tweets) and then label them correct using known fact-checking organizations because we want to watch and analyze the lifespan of a suspicious tweet and its creator. Due to the lack of such Dataset we will follow the approach of constructing our own.

In addition, all tasks that try to handle that huge amount of data, require high computation power, thus specialized tools are required as well. Such a tool which we are using in our research is Apache Spark which is a fast, in-memory tool that helps you process a great amount of data. Thus, we first need to study how those tools work, to be in place to write performant and efficient programs that will be used in our analysis.

1.3 Contributions

Our research aims to export results that will help to detect fake news disseminators on Twitter, by studying and analyzing existing fake content. Moreover, we will analyze the activity of those who spread the falsity and try to extract those user-level, content-level and network-level features that differentiate them from non-suspicious accounts. In the end, we will be able to export plots that statistically describe the most important features that separate trusted from un-trusted accounts along with a blacklist of twitter accounts, that will mention for each account a probability of disseminating fake news. Due to the lack of any Dataset that matches our needs, we will follow the approach of constructing our own Dataset of fake-suspicious tweets.

Summing up, our main contributions to the area are:

- Construct a continuous streaming dataset using Twitter crawlers.
- Export a blacklist that will contain highest fake news disseminator Twitter accounts.
- Export user-level, content-level and network-level features that differentiate suspicious from trusted accounts.

Our work will be further used together in a pipeline based on a variety of signals ranging from domain name flag-lists and natural language processing perspective to deep learning approaches with in order to create a Machine Learning model that will be used in a plugin of Check-It that tries to tackle and take a bold step towards detection and reduction of disinformation and falsity on social networks using an automated approach.

1.4 Outline Contents

Chapter 1. Introduction

In the introduction chapter, we briefly present how social networks expanded nowadays and how this is used in a bad way to lead to the problems of fake news of social networks and its serious consequences. In addition, we mention the contribution of our research to that area, what we want to export as output and how it will be used in further researches on that area. Lastly, we talk about the challenges that come across in that research.

Chapter 2. Literature & Related Work

In the second chapter, we mainly focus on analyzing literature work on fake news area and their results. Moreover, we also study and analyze other researches on similar topics such as hate speech and we explain how their work is similar to ours and how we will adjust it to our needs.

Chapter 3 Details Of Research

In this chapter, we explain in detail the process of our analysis. We explain how our system works using architecture diagrams, how we are collecting our data using our custom Twitter crawlers and how Apache Spark helps us to load them into our program. Furthermore, we give a detailed description of how we analyze our data in order to export the blacklist of Twitter accounts. In addition, we describe the process of splitting the blacklist into buckets and then using the top users of each bucket to export user-level and content-level plots and statistics. We explain all technologies, libraries and tools that are used in our analysis' programs. We then present the results of our analysis, discuss them and extract some conclusions.

Chapter 4 Evaluation

In chapter 4 we discuss the way we evaluate our results. We compare them with the literature in order to see if our user-level, content-level and network-level features of fake news disseminators are similar to the features that literature work exported. Further, analyze the number of users in each blacklist bucket that had their accounts suspended or deleted after spreading the falsity and we also use the blacklist buckets to see how the frequency of fake and overall posts is altered when the probability of fake news dissemination is increased.

Chapter 5 Future Work

In this final chapter we briefly describe how our results can be used into further analysis on fake news topic and the creation of a machine learning model that will be able to detect fake disseminators.

Chapter 2

Literature & Related Work

Contents

2.1 Fake News - Social Network View

Nowadays whether we want it or not, both true and false information spreads rapidly through online media. But falsehoods - Fake News is being diffused significantly faster, farther, deeper and more broadly than the truth in all categories of information.

Vosoughi S. in the paper "The spread of true and false news online" posted in Science [2], states that true and false statements through online media highly influences politics and economies. They mention that the spread of falsity is now a trending political strategy used to replace debates or alter stock prices and the motivation for large-scale investments. In addition, they try to examine how truth and falsity diffuse differently throughout the new social technologies and what are the factors of human and judgement which can explain these differences. Furthermore, they underline that no study has comprehensively evaluated the differences, regarding the spread of truth and misinformation yet.

Correspondingly, Potsane.M and Wai Sze L. in "Extrapolation of Aspects of Fake News on Social Networks" - [3], define fake news as news whose context has been intentionally changed and manipulated in order to influence the readers' opinions on facts of the events taking place in the real world. In addition, they underline the importance that people who live in the online world, must only receive verified and trusty news throughout social media, because misinformation can alter their choices and actions on serious situations in the real world.

2.2 Similar Researches

Besides fake news, there are also other common and modern phenomenons on social networks. Aggregation and bullying are one of them, which induces serious outcomes to victims of all demographics. In the paper "Mean Birds: Detecting Aggression and Bullying on Twitter" - [4], they try to export the features that designate abusive users on Twitter and construct a list by annotated accounts. Furthermore "Like Sheep Among Wolves: Characterizing Hateful Users on Twitter" - [5], is alike research that focuses on detecting hateful Twitter accounts. After collecting data and analyzing them they annotate most hateful accounts and also construct a list of them.

2.3 Data Collection

Data Collection in such researches is one of the most critical steps, in getting a complete and accurate picture of the area.

In the paper [5], there has been done similar research to ours, but with hateful users detection instead of fake news. In order to have more of a complete image, they sampled twitter not heavily biased towards users who used hateful words. Thus, they employed a more elaborate data collection process, which involves collecting a sample of Twitter's English speaking users and then select a subsample of these users to be annotated as hateful or not, using Twitter Streaming API.

Correspondingly, in [4], where they are trying to make research on annotating hateful users, they collected a baseline of 1 million random tweets and hate-related tweets using Twitter Streaming API.

Equivalently, another research in the literature [2] regarding Fake News, does its investigation on data containing verified true and false news stories distributed on Twitter from 2006 to 2017. Data included approximately 126.000 rumour cascades spread by approximately 3 million people more than 4.5 million times. All data were sampled from fact-checking organizations, thus they were labelled as truth or fake.

2.4 Big Data Tools - Apache Spark

In research after collecting your data, you want to analyze them following a methodology in order to export some results and statistics. Due to the fact that in such researches the amount of data we collect is huge, the traditional methods of data processing have become inefficient and time-consuming. This scale of data is called 'Big Data'. Amol Bansod in the paper "Efficient Big Data Analysis with Apache

Spark in HDFS" - [6] explains how we can avoid and defeat this problem, using Apache Spark.

Apache Spark is defined as a fast, in-memory tool that can process a great amount of data. HDFS-Hadoop Distributed File System is used by Spark to handle data which is stored by distributing over clusters. Spark then, provides high performant methods and procedures that help to analyze the data stored in HDFS in parallel.

A.Bansod - [6] then describes how Spark works and gives a detailed analysis of its architecture. Apache Spark is apart from a driver program (SparkContext), workers which you also may find them as executors, a cluster manager, and the HDFS. The driver program is the main program of spark. SparkContext is the object that gets created during the execution of a Spark program and is capable of handling the whole execution of the job. The SparkContext object connects to the cluster manager, which is used to manage the resources across clusters. Cluster managers supply executors, which are used to run the logic of the program and additionally storing the data of the application. Each application gets its own processes for the duration of the whole application and run jobs in multiple threads and needs to be network addressable from worker nodes.

Spark is mostly based on the following two concepts: Resilient Distributed Datasets (RDDs) and an execution engine which is called a Directed Acyclic Graph (DAG). RDDs are the fundamental data structure of Spark. They are an immutable distributed collection of objects and they can contain any type of Python, Java, or Scala objects including UDFs which corresponds to user-defined classes. They are also immutable once created which means that they can be transformed, or actions can be performed on them, but they cannot be changed.

Due to the fact that our data-tweets are structured which means we know the exact schema of each received tweet or we could say we have a known set of fields for each record, we are able to use Spark SQL which is Apache Spark's module for working

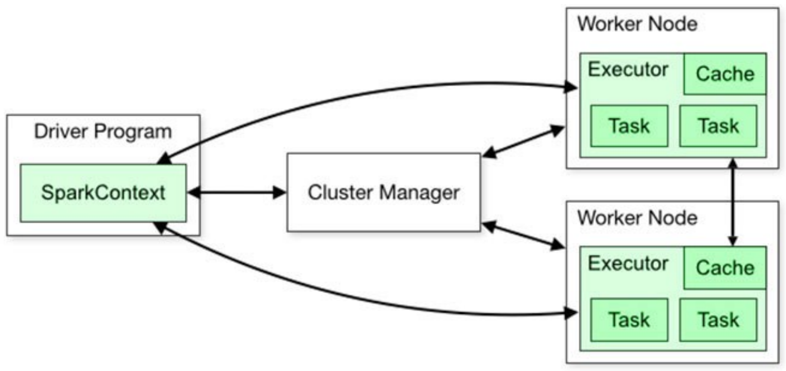


FIGURE 2.1: Apache Spark architecture

with structured data. Spark SQL stored data in DataFrames which are similar to tables in a relational database. A single DataFrame represents an RDD of Row objects. RDDs are the fundamental data structure of Spark. They are an immutable distributed collection of objects and they can contain any type of Python, Java, or Scala objects including UDFs which corresponds to user-defined classes. Due to the fact that a Dataframe knows the schema of each of its rows, data are stored in a more efficient manner than native RDDs.

Remarkable Apache Spark's advantages are the high scalability, very high processing speed and easy to API's.

2.5 Methodology

A mixed methods approach is followed in this research area, depending on what you are trying to achieve.

In Science's paper - [2] they start by collecting for each reply tweet the original tweet being replied and all the retweets of the original tweet and then using Twitter's follower graph they inferred the correct retweet path of a tweet. Each retweet cascade represented a rumour spreading on Twitter that has been labelled as true or false by the fact-checking organizations. Moreover, they proceeded to create descriptive statistics on users who participated in true and false rumour cascades and thus export vital features that differentiate fake news disseminators from truth ones.

In the literature, we are replicating from - [5] they are trying not only to export the features that differentiate hate speech disseminators but also construct a list containing the probability for each of the top hateful users, to disseminate hate speech which is what we also what our research is on but instead we want to construct a list of fake news disseminators. After sampling twitter they want to annotate the hateful users. Thus they create a lexicon of words that are mostly used in the context of hate speech and then run a diffusion process on the retweet graph, based on DeGroot's Learning Model [7]. Then they divide the users into 4 strata according to their associated beliefs after running the diffusion process and perform a stratified sampling, obtaining up to 1500 users per strata. Then analyzing the accounts and the activity of those users they export profile attributes of each user.

In [4], where they work on detecting bullying users, they initially proceed with a preprocessing on the collected data. They clean the data that have noise in them, i.e remove numbers, stop words and punctuations, as well as converting all characters to lower case. In addition, they remove spammers from the dataset. In order to check whether a user is spamming they rely on two main indicators of spam: the user is using a large number of hashtags in his tweets, and the user is posting a large number of tweets highly similar to each other. Moreover, they label users as aggressor, bully or spammer according to their activity. Following a session-based approach, they divide their collected data into sliding windows and they then proceed into a feature extraction process which consists of user-based, text-based and network-based features.

2.6 Results

In Science's paper - [2] results showed that falsehood diffuses significantly farther, faster, deeper and more broadly than the truth in all categories of information. In addition, when they compared users' Twitter accounts who were involved in true and false rumour cascades, statistics and metrics indicated that possible fake news disseminators had significantly fewer followers, followed compelling fewer people. Moreover, false advertisers appeared to be far less active and their accounts' age was undoubtedly shorter. This, however, is reasonable due to the fact that once fake posts are labelled by fact-checking organizations, owner's accounts are either suspended or deleted.

Besides in [3], they also exported similar descriptive statistics. Equivalently, fake news disseminators' accounts had less number of followers in comparison to clean accounts. Additionally, their accounts lasted not long and appeared to follow more people on Twitter. However, user account characteristics alone cannot explain much about posts on social networks, thus they also export statistics regarding the content of fake posts. Fake tweets appeared with far more hashtags used along with external links/URLs to the original source of the news. Moreover, fake posts had special characters like exclamation marks and question marks.

After running feature selection and feature importance methods they created a table of the most prominent features which were found to be the most considered for the purpose of detecting fake news on social networks. Some of the important user-based features were the number of followers, followees and account's age. On the other hand, most prominent post features were the number of hashtags and the number of external URLs.

Chapter 3

Details Of Research

Contents

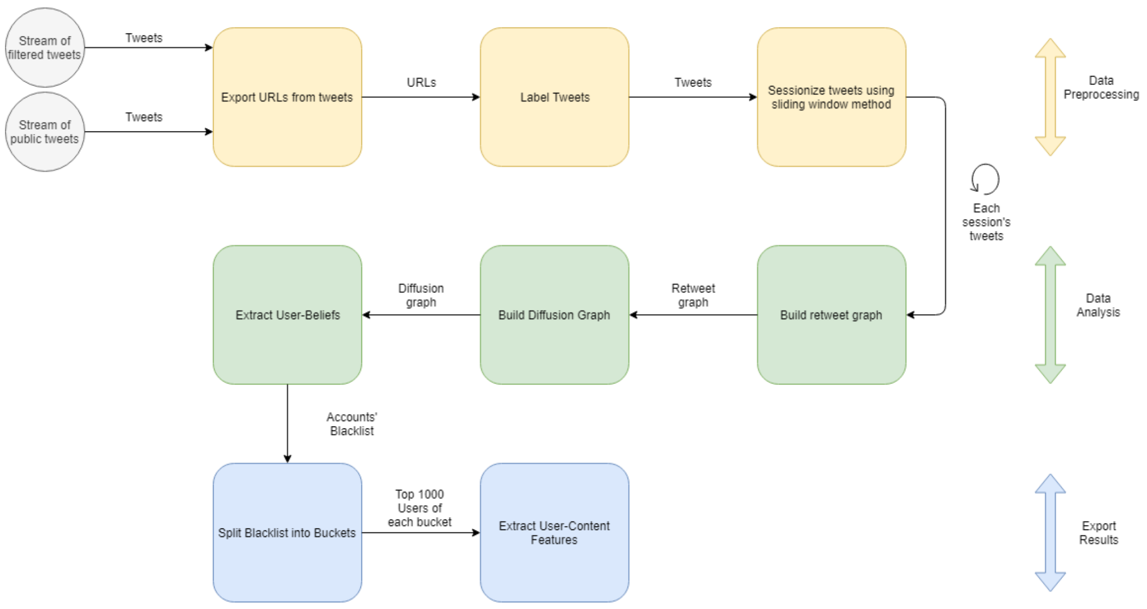


FIGURE 3.1: Methodology Architecture

3.1 Architecture

In order to have a better view of our methodology, we start by constructing an architecture diagram, that provides an abstract explanation of our analysis' steps and also gives a brief description of what we are trying to achieve.

Initially, we want to collect or find a dataset that will contain both fake and real tweets. Following, we will proceed by extracting all the extended URLs from each tweet's entities. Then, we take those URLs' list and we want to decide which of those are really fake so that we can properly label the corresponding tweet that contained each of those URLs.

Up at this point, we have a list of labelled tweets as fake or real. Moreover, we take that list and feed it into a Spark program that will split the tweets into sessions using the sliding window method based on each tweet's timestamp-ms field. We then loop for each on all sessions. For each session, we construct the retweet graph and then the diffusion process-graph using DeGroot's model [8], which gives us the blacklist

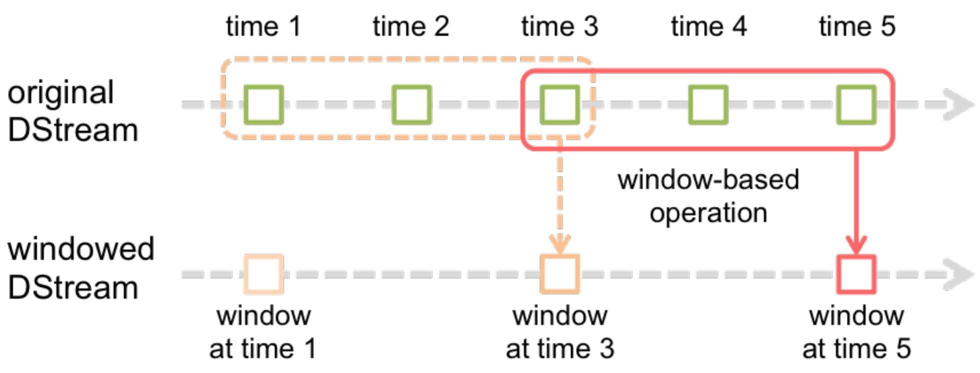


Figure 3.2: Sliding Window Method

that contains the found Twitter accounts along with their fake news dissemination probability.

After processing all sessions, we have a full blacklist of Twitter accounts. We split the users into buckets using the belief intervals [0,0.25).[0.25,0.50),[0.50,.75),[0.75,1.0) and get the top 1000 active users of each bucket for further activity analysis. By active users we mean the users who appeared the most in our sessions as this would give a more overall results. Lastly, from the activity analysis, we extract the user and content level features and plot them in order to see how high-probability disseminators are differentiated from low-probability ones.

3.2 Data Collection

Data collection in such researches is always one of the most vital procedures for efficient and meaningful analysis.

As we are trying to characterize and annotate fake news disseminators, it would not be appropriate to only collect fake tweets as this technique would bias our analysis and not produce valid results. Thus, we want to find a dataset that also includes sample tweets except for fake ones. We also want to find a continuous dataset, meaning that we want a dataset that would include tweets that were collected from a continuous crawling because this would mean that our analysis and results will be more quantitative, as we would be able to track the lifespan and activity of fake news disseminators after posting a fake tweet.

However, after a tough search for a dataset that would meet our needs, unfortunately, we did not find any. Thus, we proceed into constructing our own dataset. As it is described by the architecture diagram 3.1 we start by setting up a Virtual Machine that will run two Java Programs which will be our Tweet crawlers. We use the Twitter Streaming API to crawl tweets.

The first crawler will just sample from Twitter, thus we do not pass any keywords that we from the stream to track. We simply want to query only English tweets due to the fact that later we want to extract content features using natural language processing libraries that work mostly on English text. The second crawler won't just sample Twitter but we need to track tweets that contain suspicious URLs. Thus, we pass a query to the crawler that tracks keywords from a list. By suspicious URLs we mean a list we built after searching for the most suspicious domains that often host fake news. It is essential to clarify that we do not know for sure that those URLs actually post fake news, we just know that there is a high probability of it. That way, the crawler will detect tweets that contain any of those suspicious URLs-keywords.

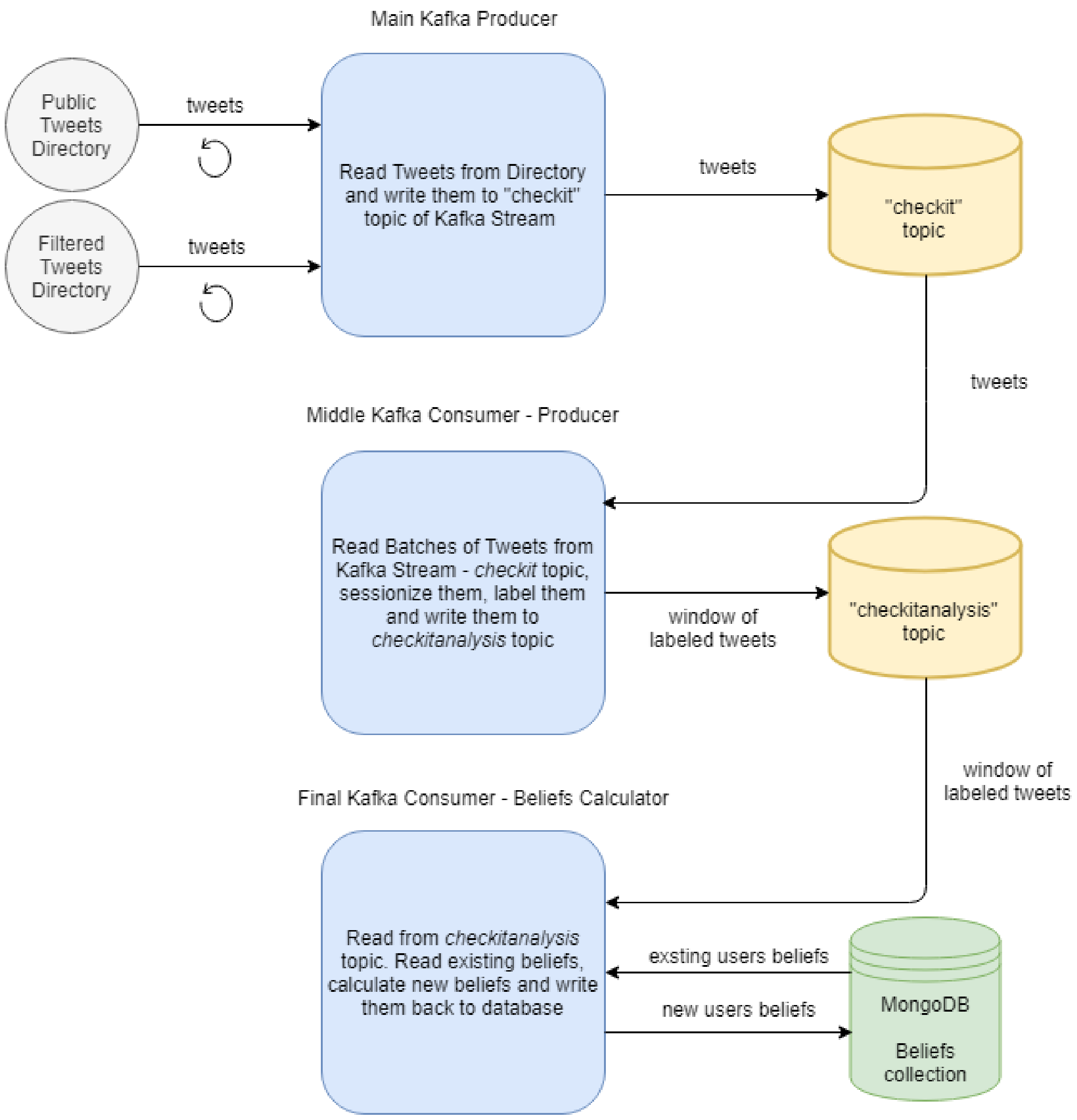
Both crawlers were started at the same time as two separated programs. We store the received tweets as zipped multiline json-files. Crawlers were running for about a month and collected approximately 150 million tweets, with the 30 million being suspicious and 120 million public.

3.3 Data Analysis

3.3.1 Load Collected Data

After collecting an efficient amount of data we proceed in our analysis. We want initially to create a program that will be able to load all data and process them. As discussed before we will be using Apache Spark a tool that can process a huge amount of data. Using Spark SQL we can read all the zip files from a directory that contains two subdirectories which are the collected public and fake tweets. All data are loaded in a Dataframe.

However that approach because of the size of data, proved to quite a resource costly and discouraged. Thus, since we do not have sufficient hardware to be able to support this approach we decide not to load all the data at once but small amounts at a time in a stream as batches following a different approach as shown in Figure 3.3 .In order to support this method, we need to first write a program that will read the collected data and write them in batches into a stream. This program is called Kafka Producer. Apache Kafka [9] is a distributed streaming platform that is used for building real-time data pipelines and streaming apps. We use Kafka Producer API to build a program that will be reading continuously from both fake and public directories and publish tweets in a stream. We also use the Kafka Consumer API, to write a middle program that will consume the data from the first topic of the stream, session them and label them as probably fake or not according to a suspicious domains list and then writes them again into another topic on the stream to be further analyzed by another program. The final consumer is a program that will consume each window of labeled tweets. It will then, read from the database the existing user beliefs, calculate the new ones and then update them into the database. We also lower the retention interval of the stream so that we do not have a large



Main Kafka Producer
Public Tweets Directory
Filtered Tweets Directory
tweets
tweets
Read Tweets from Directory and write them to "checkit" topic of Kafka Stream
tweets
"checkit" topic
tweets
Middle Kafka Consumer - Producer
Read Batches of Tweets from Kafka Stream - *checkit* topic, sessionize them, label them and write them to *checkitanalysis* topic
window of labeled tweets
"checkitanalysis" topic
window of labeled tweets
Final Kafka Consumer - Beliefs Calculator
Read from *checkitanalysis* topic. Read existing beliefs, calculate new beliefs and write them back to database
exsting users beliefs
new users beliefs
MongoDB
Beliefs collection

FIGURE 3.3: Spark-Kafka Data-Load Architecture

amount of data left on the stream since we do not need them after they are consumed from the consumer.

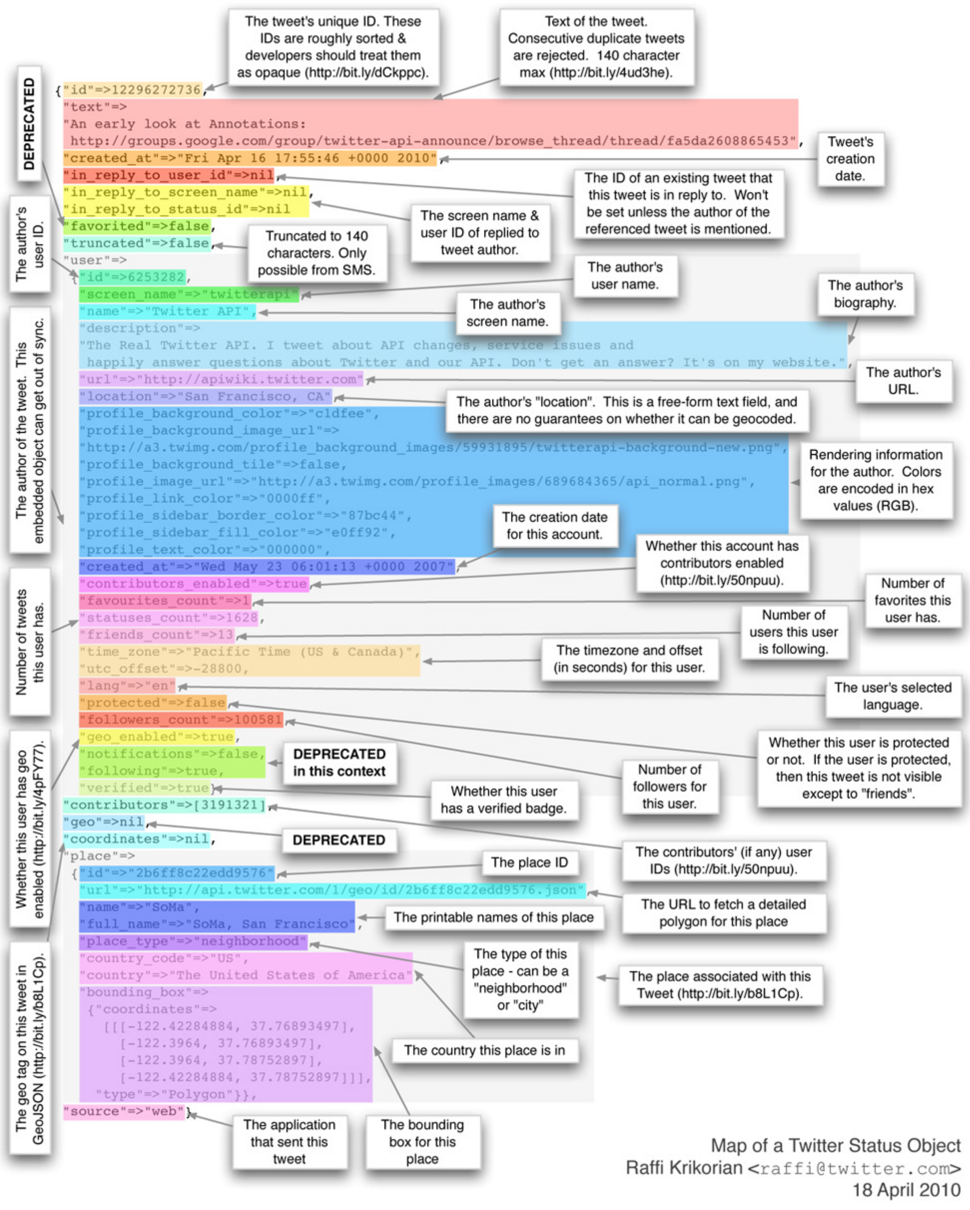
3.3.2 Sessionization

Since we do not want to analyze single tweets because it does not provide enough context to discern whether a user is actually disseminating fake news or not, we want to group tweets into time clusters which are called sessions. In order to achieve that we are using Spark's built-in window function as shown in figure 3.2 which allows us to gather and group tweets into windows. Window function expects from us to pass the field of the object which will be taken into account when creating the windows. In our case, we are passing the creation timestamp field which we get from each tweet's object. We also have to pass the desired window duration along with sliding duration which indicates the size in time that the window will be moved for the next one. At this point, it is important to mention that a tweet received from Twitter Streaming API comes in a JSON format and consists of multiple metadata as show in Figure - .

3.3.3 Retweet Graph

We define fake news disseminator, a user who not only posts fake news but also influences others to share his post which means he affected them. Retweet graphs have been widely used in the social network research area, with previous works proposing that retweets are better than followers to judge the influence of users [10]. Therefore, we want to create the retweet-induced graph which illustrates how a user who posts fake tweets influences his neighbours in his opinion. The retweet-induced graph is defined as a directed graph G = (V, E) where each node u V represents a user in Twitter and each edge (u1,u2) E represents a retweet in the network, where the user u1 has retweeted user u2.

For our case, in order to create graphs, we use NetworkX a python's library which is used for studying graphs and networks. As shown in Figure 3.5 after receiving



```
{"id"=>12296272736,
"text"=>
"An early look at Annotations:
 http://groups.google.com/group/twitter-api-announce/browse_thread/thread/fa5da2608865453",
"created_at"=>"Fri Apr 16 17:55:46 +0000 2010",
"in_reply_to_user_id"=>nil,
"in_reply_to_screen_name"=>nil,
"in_reply_to_status_id"=>nil
"favorited"=>false,
"truncated"=>false,
"user"=>
 {"id"=>6253282,
 "screen_name"=>"twitterapi",
 "name"=>"Twitter API",
 "description"=>
 "The Real Twitter API. I tweet about API changes, service issues and
  happily answer questions about Twitter and our API. Don't get an answer? It's on my website.",
 "url"=>"http://apiwiki.twitter.com",
 "location"=>"San Francisco, CA",
 "profile_background_color"=>"c1dfee",
 "profile_background_image_url"=>
 "http://a3.twimg.com/profile_background_images/59931895/twitterapi-background-new.png",
 "profile_background_tile"=>false,
 "profile_image_url"=>"http://a3.twimg.com/profile_images/689684365/api_normal.png",
 "profile_link_color"=>"0000ff",
 "profile_sidebar_border_color"=>"87bc44",
 "profile_sidebar_fill_color"=>"e0ff92",
 "profile_text_color"=>"000000",
 "created_at"=>"Wed May 23 06:01:13 +0000 2007",
 "contributors_enabled"=>true,
 "favourites_count"=>1,
 "statuses_count"=>1628,
 "friends_count"=>13,
 "time_zone"=>"Pacific Time (US & Canada)",
 "utc_offset"=>-28800,
 "lang"=>"en",
 "protected"=>false,
 "followers_count"=>100581,
 "geo_enabled"=>true,
 "notifications"=>false,
 "following"=>true,
 "verified"=>true},
"contributors"=>[3191321],
"geo"=>nil,
"coordinates"=>nil,
"place"=>
 {"id"=>"2b6ff8c22edd9576",
 "url"=>"http://api.twitter.com/1/geo/id/2b6ff8c22edd9576.json",
 "name"=>"SoMa",
 "full_name"=>"SoMa, San Francisco",
 "place_type"=>"neighborhood",
 "country_code"=>"US",
 "country"=>"The United States of America",
 "bounding_box"=>
  {"coordinates"=>
    [[[-122.42284884, 37.76893497],
      [-122.3964, 37.76893497],
      [-122.3964, 37.78752897],
      [-122.42284884, 37.78752897]]],
   "type"=>"Polygon"}},
"source"=>"web"}
```

The tweet's unique ID. These IDs are roughly sorted & developers should treat them as opaque (http://bit.ly/dCkppc).
Text of the tweet. Consecutive duplicate tweets are rejected. 140 character max (http://bit.ly/4ud3he).
DEPRECATED
Tweet's creation date.
The ID of an existing tweet that this tweet is in reply to. Won't be set unless the author of the referenced tweet is mentioned.
The screen name & user ID of replied to tweet author.
Truncated to 140 characters. Only possible from SMS.
The author's user ID.
The author's user name.
The author's screen name.
The author's biography.
The author's URL.
The author's "location". This is a free-form text field, and there are no guarantees on whether it can be geocoded.
Rendering information for the author. Colors are encoded in hex values (RGB).
The author of the tweet. This embedded object can get out of sync.
The creation date for this account.
Whether this account has contributors enabled (http://bit.ly/50npuu).
Number of favorites this user has.
Number of tweets this user has.
Number of users this user is following.
The timezone and offset (in seconds) for this user.
The user's selected language.
Whether this user is protected or not. If the user is protected, then this tweet is not visible except to "friends".
Number of followers for this user.
DEPRECATED in this context
Whether this user has a verified badge.
Whether this user has geo enabled (http://bit.ly/4pFY77).
The contributors' (if any) user IDs (http://bit.ly/50npuu).
DEPRECATED
The place ID
The URL to fetch a detailed polygon for this place
The printable names of this place
The type of this place - can be a "neighborhood" or "city"
The place associated with this Tweet (http://bit.ly/b8L1Cp).
The country this place is in
The geo tag on this tweet in GeoJSON (http://bit.ly/b8L1Cp).
The application that sent this tweet
The bounding box for this place
Map of a Twitter Status Object
Raffi Krikorian <raffi@twitter.com>
18 April 2010

Figure 3.4: Tweet Metadata

a window of tweets we want to create the retweet-induced graph. Looping then, through all received tweets we first add a self-loop to each user who appeared in the window due to the fact that a user not only affects other users who retweet his post but also affects himself. We know whether a tweet is retweeted or not, from the retweeted boolean field that is included in a tweet's object. In case a suspicious

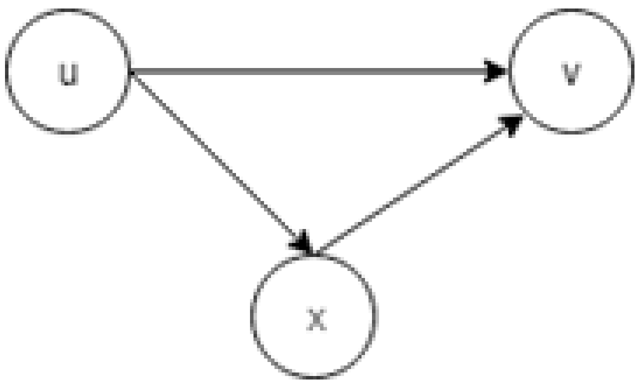


FIGURE 3.5: Retweet-graph example. U retweeted x and v. X retweeted V

tweet is retweeted we add to the directed graph an edge from the retweeted user to the origin user. As data on the edge, we want to hold the id of the tweet that connects the two nodes-users.

3.3.4 Diffusion Graph - Process

Following the literature work, we are replicating "Characterizing Hateful Users on Twitter", after constructing the retweet graph a diffusion process follows up. Our diffusion process is based on DeGroot's Learning Model [7]. DeGroot's model is used since it introduces a simple mechanism of opinion propagation: every individual forms shares her opinion by averaging her own opinion with those of her friends.The process is repeated until all opinions converge. Although the mechanism is simple, it models sufficiently opinion diffusion and incorporates elaborate characteristics of the process

Initially, we want to build the infected graph that results after annotating the suspicious users in our retweet graph, which means the Twitter accounts that posted a tweet containing at least one of the suspicious domains. Then, we run the diffusion process which can be described as follows:

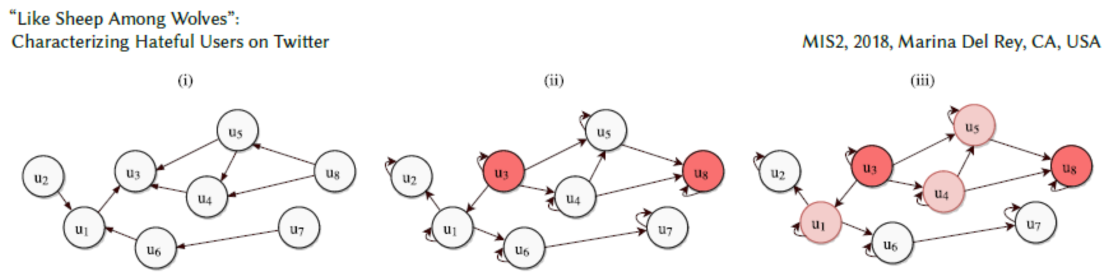

"Like Sheep Among Wolves":
Characterizing Hateful Users on Twitter
MIS2, 2018, Marina Del Rey, CA, USA
(i)
(ii)
(iii)
u1
u2
u3
u4
u5
u6
u7
u8

FIGURE 3.6: i) Retweet Graph ii) Infected Graph iii) Diffusion Graph. Red nodes are the users that posted a suspicious tweet in the current window. Pinks are those that were infected

Let A be the adjacency matrix of the retweeted-induced graph G=(V, E) where each node u V represents a user and each edge (u,v) E represents a retweet. We have that A(u,v) = 1 if u retweeted v. We create a transition matrix T by inverting the edges in A (as the influence flows from the retweeted user to the user who retweeted him or her), adding a self-loop to each of the nodes and then normalizing each row in A so it sums to 1 (This means each user is equally influenced by every user he or she retweets) as it is shown in figure 1.4. Matrix T includes the weight a node adds on anothers opinion (fake news dissemination). We then associate a belief pi(0) = 1 to every user who posted a suspicious tweet and pi(0)= 0 to all who did not. Lastly, we create new beliefs p(t) using the updating rule:

$$p(t) = Tp(t1) \tag{3.1}$$

Therefore, if a user who has posted or retweeted a suspicious tweet (containing a suspicious URL from the domains blacklist) and has eventually influenced more users with his opinion by retweeting him, then the veracity score of that user is decreased because it has propagated and disseminated a fake tweet.

3.3.5 Storing Beliefs

After creating retweet and infected graph and calculate the window's diffusion we have a new set of Twitter accounts along with their beliefs. We want to store each window's beliefs into a database, in order to be able to proceed into further analysis and extract some features of the top fake news disseminators. For the purposes of this research, we store our data in a non-relational database, MongoDB. In a collection, we store all found Twitter account's beliefs along with some extra information including the screen name, the id and the total appearances which indicate the total number of windows an account appeared in. We are using MongoDB Spark Connector which is a package that provides integration between MongoDB and Apache Spark.

Initially, we retrieve the collection of the beliefs into a DataFrame. Afterwards, we make a full outer join on that DataFrame and the new-created beliefs Dataframe based on the id and the name of each row (Twitter account). Next, we increase by 1 the total appearances of the accounts that already existed in the database and recalculate the beliefs using the following average calculator formula:

$$totalBelief = (existingBelief * (totalAppearances - 1)) + newBelief \tag{3.2}$$

$$newAverageBelief = totalBelief/totalAppearances \tag{3.3}$$

Finally, we are writing back to the database the newly calculated beliefs. That way, we update the belief of the existing accounts and create a new record for the first time seen accounts.

Note that all the aggregation functions we execute on DataFrames' rows include some custom logic based on our needs and Spark does not provide any built-in functions that we can use. However, it gives us the option to implement our own functions that can include any logic we want. Those functions are called UDF - User Defined Functions.

3.3.6 Feature Extraction

After constructing the users' blacklist we also want to study how fake news disseminators are differentiated from trusted ones. To do so, we follow a similar approach as in the literature [5]. We start by loading the whole users' beliefs collection-blacklist from the database into a DataFrame. We then want to split the users into four buckets based on their belief. We can accomplish that by using Spark's built-in filter function on the main DataFrame. Thus we distribute the users into four buckets with the belief intervals [0,.25),[.25,.50),[.50,.75),[.75,1). As a result, we have now 4 DataFrames.

Following up, we take the top 1000 users of each bucket-DataFrame using Spark's limit function, as that would be a representative sample of users to analyze. For each user in all buckets, we want to extract some information from his Twitter account and also track his history which means to retrieve a number of his last posts. We are using Tweepy - an easy to use Python library for accessing the Twitter API - to help us get information about a Twitter account. Since we have the id of each Twitter account we can easily query Tweepy to get a user's Twitter object. From each Twitter account, we want to focus on certain fields-features that we saw from the literature, that help us to detect fake news disseminators.

In addition, as we comprehend from the literature, user attributes alone cannot reveal much about fake news propagation, therefore we also want to study the content of the posts of those suspicious fake news accounts. Tweepy helps us to achieve that

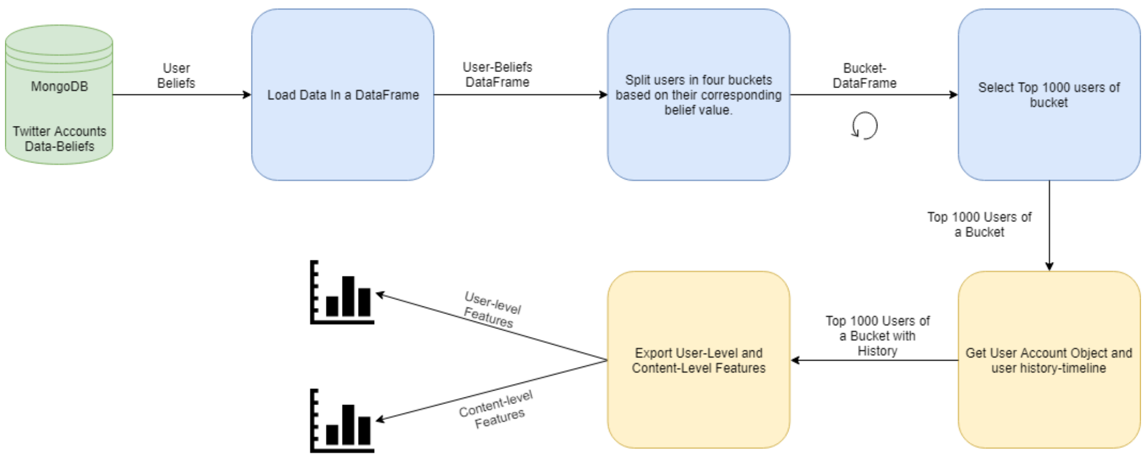

MongoDB
Twitter Accounts Data-Beliefs
User Beliefs
Load Data In a DataFrame
User-Beliefs DataFrame
Split users in four buckets based on their corresponding belief value.
Bucket-DataFrame
Select Top 1000 users of bucket
Top 1000 Users of a Bucket
Get User Account Object and user history-timeline
Top 1000 Users of a Bucket with History
Export User-Level and Content-Level Features
User-level Features
Content-level Features

FIGURE 3.7: Feature extraction process

since it provides a function to read an account's timeline-history. Furthermore, we are free to choose the number-depth of past tweets of each account. We choose it to be 3000 as we believe that it is a representative number to examine a user's history timeline and extract some inferences.

We present the above procedure of tracking suspicious user's activity as shown in Figure - 3.7

3.4 Results

3.4.1 User's Blacklist

As a result of the diffusion process, we end up having stored approximately 8 million users in our database. Out of those users 419 end up with belief between 0.75 and 1. 16028 accounts have belief between 0.5 and 0.75. 13060 accounts have belief between 0.25 and 0.5 and lastly, 77267 accounts have belief between 0 and 0.25. The rest of them who end up with zero belief are accounts who did not post any suspicious tweet. For every document-user in our database we store the Twitter account's id, name the corresponding belief and the total number of appearances which the user appeared in our windows. The whole collection where beliefs are stored ends up with approximately 1.8G in total size.

3.4.2 Fake News disseminators' features

We further investigated the most relevant attributes of fake news on social networks. After studying the literature we want to focus on specific features that are believed to differentiate fake news disseminators and so help in detecting fake news on social networks and so Twitter. We select the most important and relevant user-level features - Table 3.1 and content-level features - Table 3.2 for our research.

User-level features
Average User Friends
Average User Followers
Average Users With Description
Average Verified Users
Average Account Age (in days)

TABLE 3.1: User-level Features

The user-level features we are focussing on are the number of followers and friends (follow and being followed from an account) of an account, whether the user is verified or not, the number of posts that a suspicious account is posting, whether the user has written a description and the existence age of the account [11]. As for the content-level features of suspicious posts, we are concentrating on the number of exclamation and question marks of the text, the number of mentions, URLs and hashtags, the number of first-person pronouns and the sentiment polarity (emotions

Content-level features
Average content exclamation marks
Average content question marks
Average content length
Average content URLs
Average content Sentiment Polarity

Table 3.2: Content-level features

expressed from content, can be positive or negative) of the content and lastly the length of the text.

Statistics plots are constructed after extracting the features for the top 1000 Twitter accounts of each bucket as described before. We present and compare the same feature for all buckets. Important to note, is that the higher the belief a user has, the higher the probability is to disseminate fake news. In order to visualize our results we use bar-charts and box-blots. Bar chart is a plot that presents categorical data with rectangular bars with heights or lengths proportional to the values that they represent. Due to the fact that in our results we detect a lot of outliers we also choose to visualize our results in box-plots. A box plot, also called a box-and-whisker plot, is a chart that graphically represents the five most important descriptive values for a data set. These values include the minimum value, the first quartile, the median, the third quartile, and the maximum value. That type of plots can help us detect and visualize noisy data in our dataset that simple bar charts cannot.

Initially, our results showed that ultra-high belief users had significantly fewer friends on their accounts than low-belief ones - Figure 3.8. Followers of that tier accounts seems to be following in the same scale - Figure 3.9.

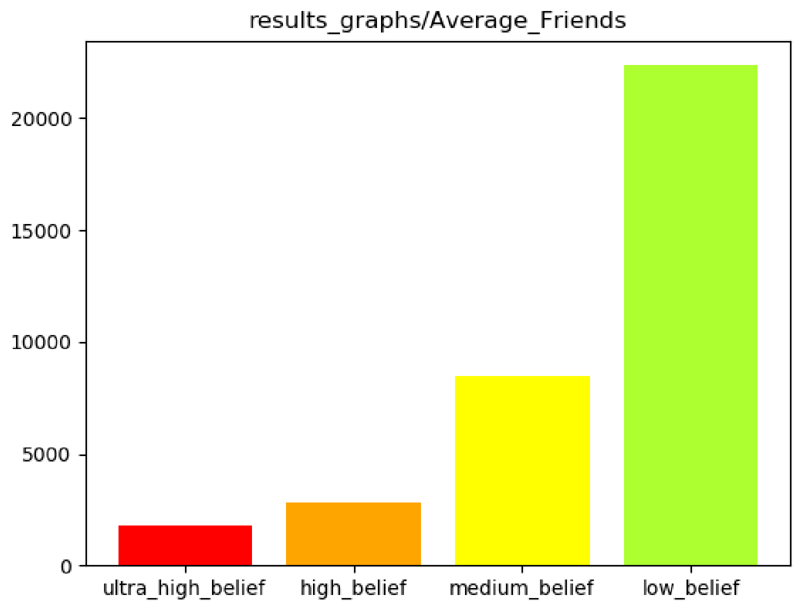


FIGURE 3.8: Average User Friends

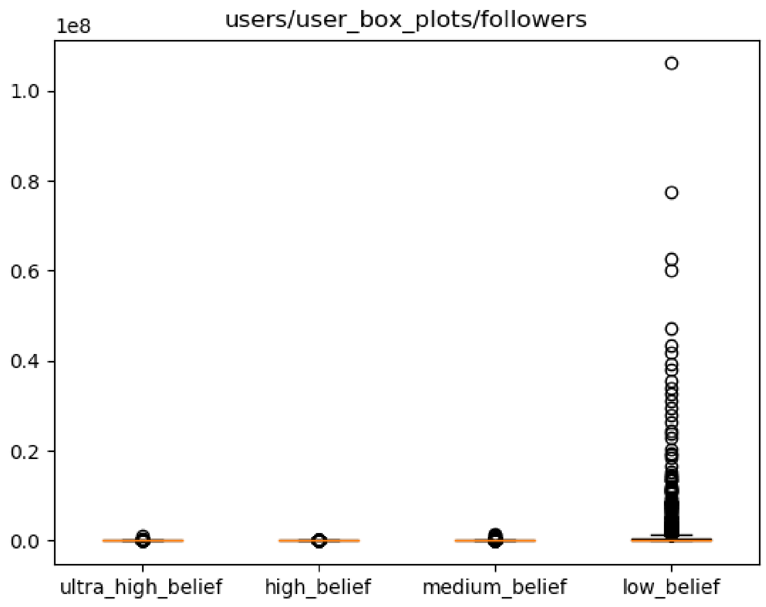


FIGURE 3.9: Average User Followers

In addition, the percentage of suspicious users who had a description on their account appeared to be shorter than more trusted users - Figure 3.10.

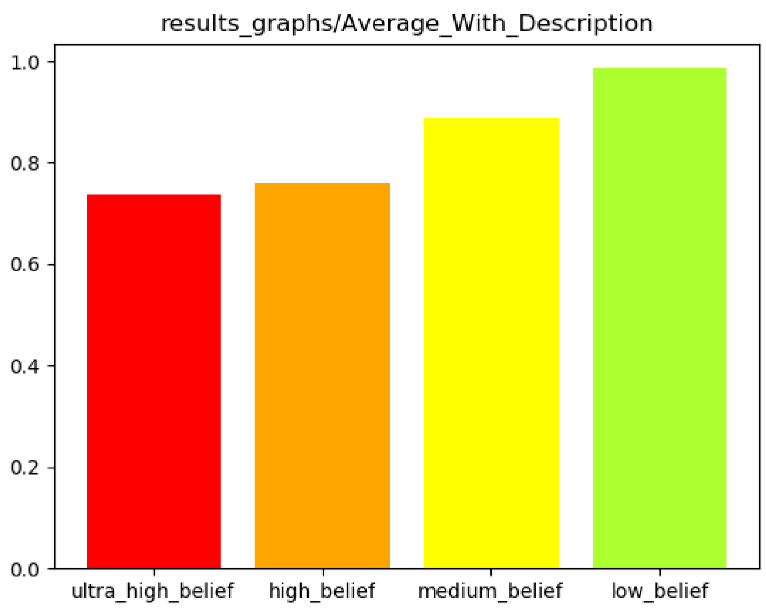


FIGURE 3.10: Average Users With Description

Another important statistics we extract is that accounts which spread fake news on social networks do not seem to last long because they are quickly removed. That can be noticed by Figure 3.11 which indicates that fake news disseminator accounts have a fewer number of days of existence in comparison to cleaner accounts. Supporting on that result it is remarkable that from the 4000 users we tracked (1000 for each of the 4 buckets), approximately 290 of them appeared to have their accounts deleted or suspended.

Moreover suspicious users were much significantly fewer times verified - Figure 3.12. Finally, we see that users with incremental belief are posting more fake posts that more trusted users - Figure 3.13. According to our approach that means that in their posts they often use URLs that come from suspicious-untrusted domains which are likely to be posting fake articles-posts.

Moving on to content-level features we notice that top tier fake news posts have a bit less exclamation - Figure 3.14 and question marks - Figure 3.15 but the whole

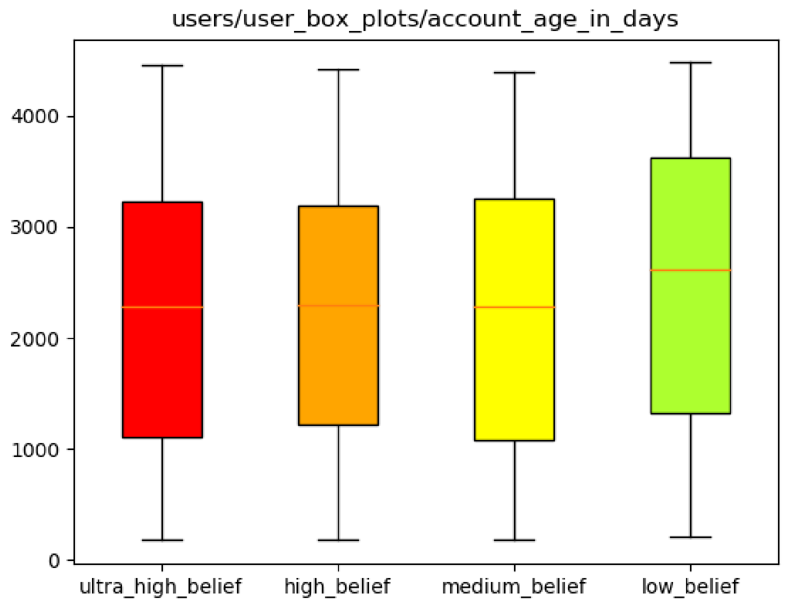


FIGURE 3.11: Average Account Age

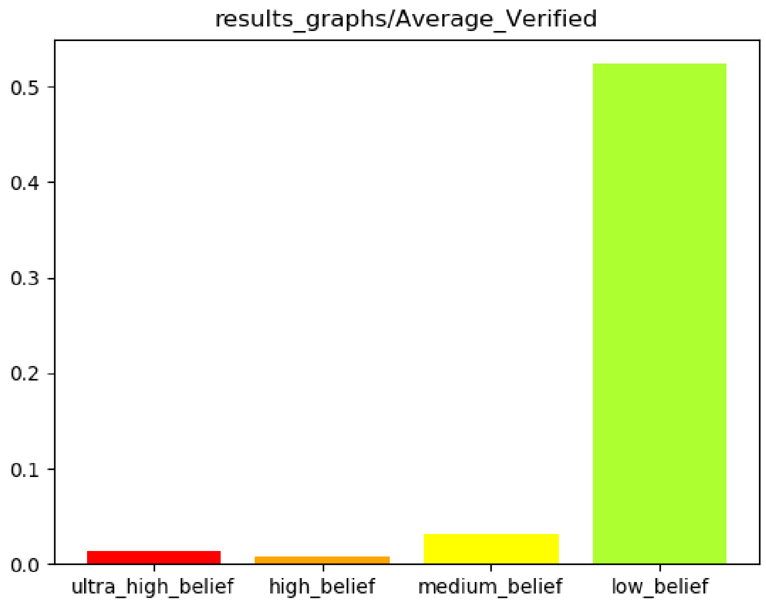


FIGURE 3.12: Average Verified Users

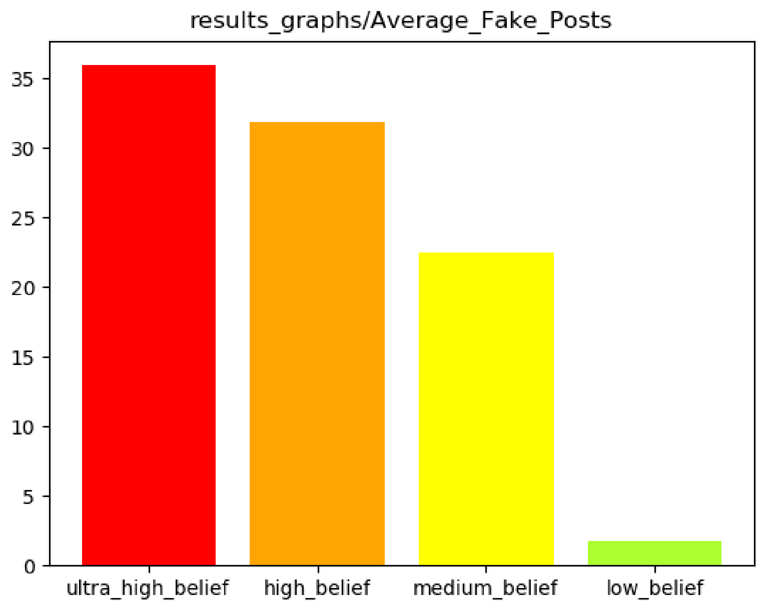


Figure 3.13: Average Fake Posts

length of the content appeared to be quite on the same scale for all the buckets - Figure 3.16.

For the entities object of a suspicious tweet, we view that ultra-high belief users' posts had fewer mentions but significantly more URLs - Figure 3.17.

We believe that is an expected behaviour of fake posts as they want to inspire others to click on external links that redirect to probably a fake post. In addition the overall sentiment polarity of most suspicious tweets appeared to be significantly lowered then more trusted ones - Figure. That shows that suspicious tweets express more often a negative emotion. 3.18.

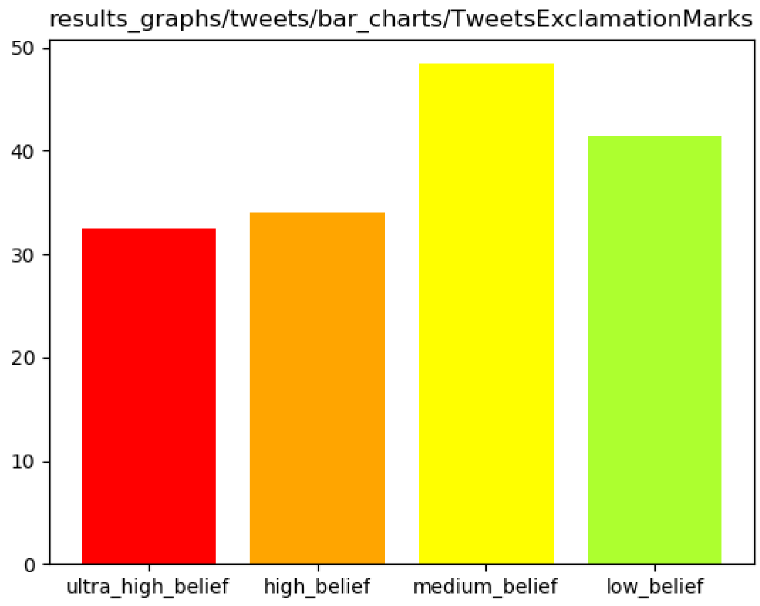


Figure 3.14: Content Exclamation Marks

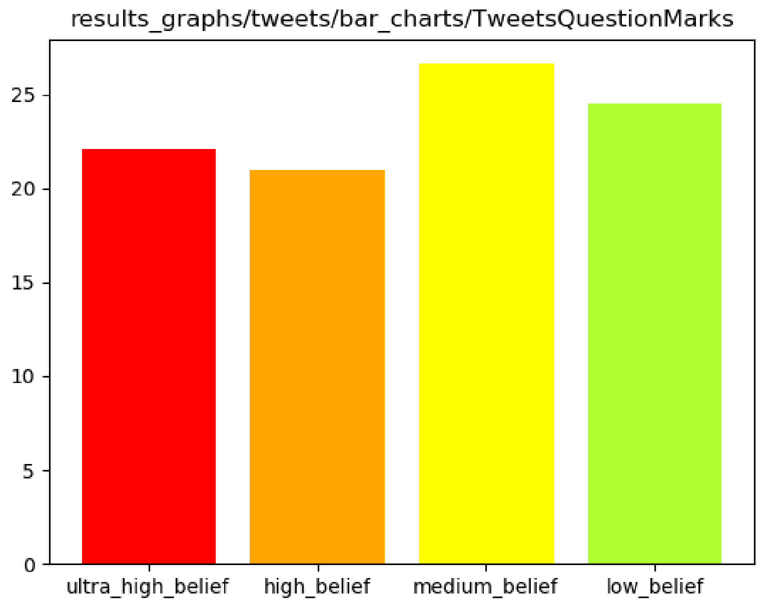


Figure 3.15: Content Question Marks

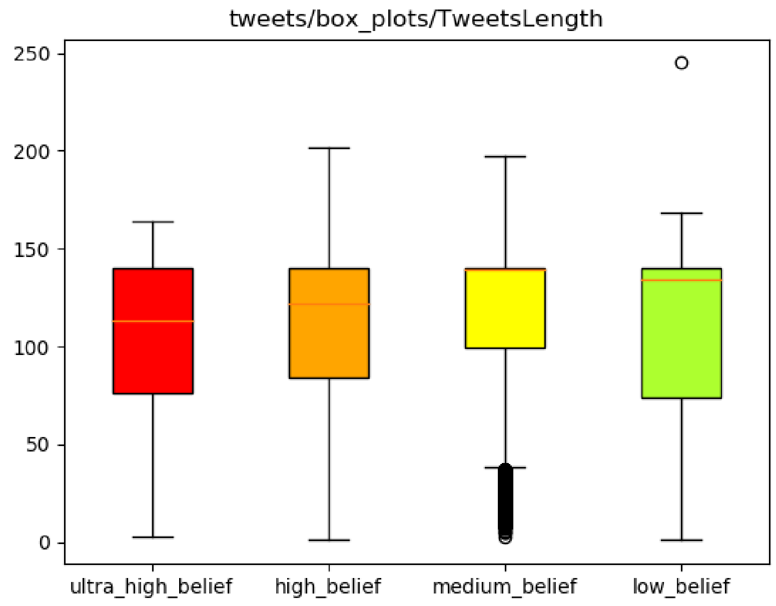


FIGURE 3.16: Average Content Length

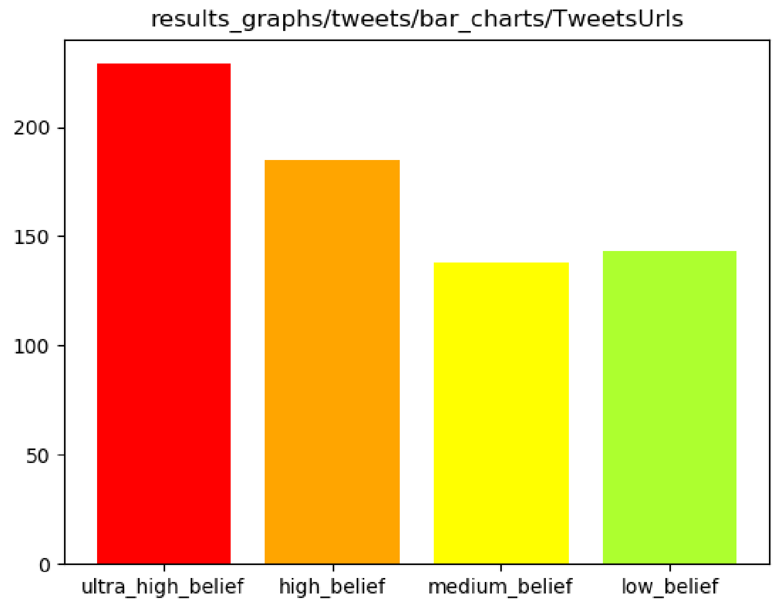


FIGURE 3.17: Average Content URLs

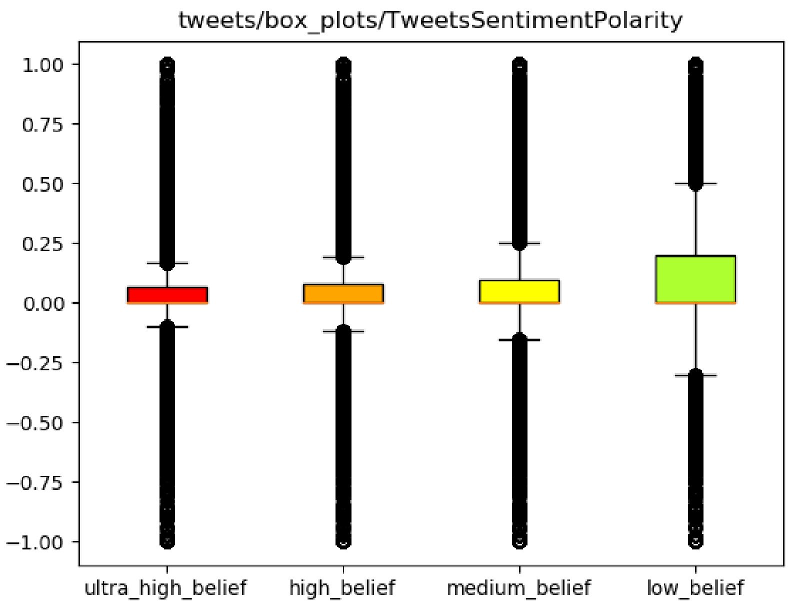


Figure 3.18: Average Content Sentiment Polarity

Chapter 4

Evaluation

Contents

4.1 Feature Evaluation

To evaluate our results and concludes that result from the plots, we compare them with literature work. As we see, similar results were exported in Science's research [2]. Their results reveal that users who spread false news had significantly fewer followers and were significantly less active. Furthermore, fake news disseminator accounts were verified less often and had been on Twitter for significantly less time. Moreover and one of the most important features is the account age of those suspicious accounts. Science's research, as our research showed that fake news disseminators seem to be on Twitter for significantly less time because their accounts are getting initially suspended when detected of posting fake news and eventually deleted if continuing the spread of falsity. That theory is also extracted from another literature work [3].

4.2 Blacklist Evaluation

Our user blacklist seems to be quite positive as well. As we know, fake news disseminators are very possible to have their accounts suspended or deleted after disseminating the falsity. Thus, we take the 4 belief-buckets and take the top 1000 accounts of each bucket to check the status of each of those accounts. In order to check the status of an account, we use again Tweepy. We simply query Tweepy passing the user's account id (that we have already stored in our database) to get a user's Twitter object. If the account is still active on Twitter Tweepy will act as normal and return as the Twitter's account as a JSON object. If now, for whatever reason this fails, Tweepy throws an error object that contains the exact error code as an integer value and the error message. According to the error code, we decide whether the account is suspended (error code is 63) or deleted (error code is 50 - will throw a 'user not found' message) as shown in Figure 4.2. In order to revalidate that approach, we can simply visit Twitter in our browser and we will see a page that tells us whether the account is in fact suspended or deleted. Our results as show in figure 4.1 tend to show that higher-probability fake news disseminators have their account suspended or deleted more often than less suspicious ones.

In addition, when querying most recent 3000 tweets of each one of the top 1000 users of each belief-bucket we plot their daily overall tweeting frequencies and then compared them with the frequencies of suspicious tweets containing URLs of known fake news domains. To do so, we start by loading from our database the 4 buckets which contain the users with their corresponding beliefs along with their history. Due to the fact that a user may be active only a certain amount of days and not all days of the given timeline, we do not choose to just find the average daily activity

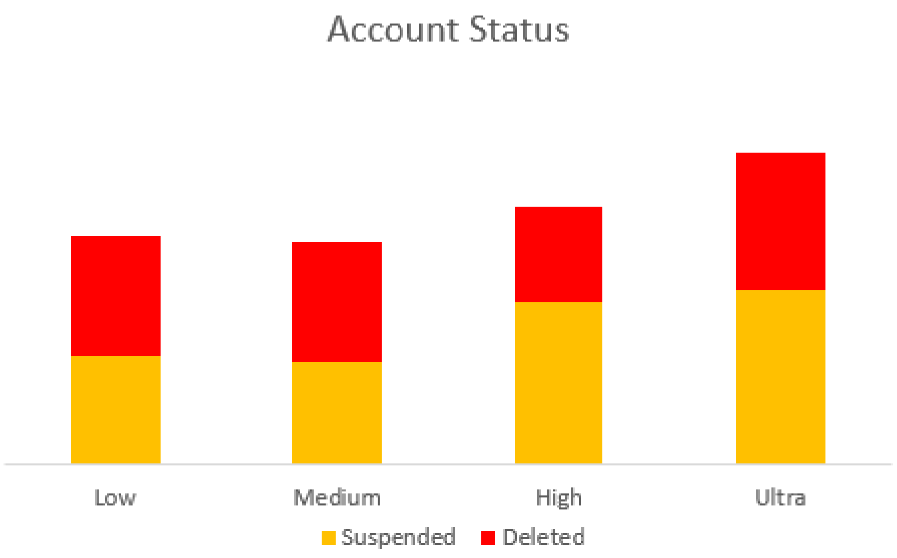
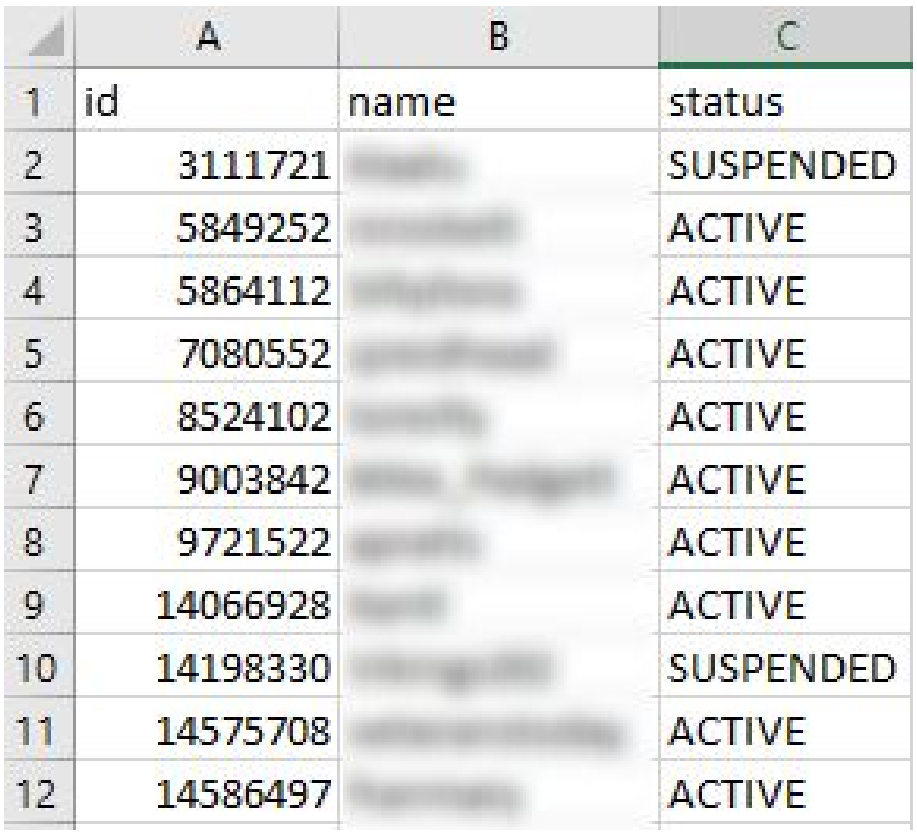


FIGURE 4.1: Number of deleted-suspended accounts per bucket



	A	B	C
1	id	name	status
2	3111721	[redacted]	SUSPENDED
3	5849252	[redacted]	ACTIVE
4	5864112	[redacted]	ACTIVE
5	7080552	[redacted]	ACTIVE
6	8524102	[redacted]	ACTIVE
7	9003842	[redacted]	ACTIVE
8	9721522	[redacted]	ACTIVE
9	14066928	[redacted]	ACTIVE
10	14198330	[redacted]	SUSPENDED
11	14575708	[redacted]	ACTIVE
12	14586497	[redacted]	ACTIVE

FIGURE 4.2: Sample of Detecting status of suspicious-blacklisted Twitter accounts

but the actual-real daily activity. In order to accomplish that, we loop through all the history timeline of each user and each time we save in a dictionary the date of the creation of the tweet, which can be easily accessed from the created_at attribute which each tweet object holds. By the end of that process, we end up with a dictionary whose keys are the active days of each account (the actual dates when posts were created). We are at this point able to plot the overall tweet frequency of each of the buckets.

As we can see from the plots, low-belief users have a large overall tweeting frequency with a low frequency of tweets with fake content which is was expected since in that bucket belief ranges from [0,.25) which means that we also have users that have zero belief and never have posted a tweet containing suspicious URLs - Figure 4.3. Medium-belief users appear with a small rise in the frequency of tweets containing fake news articles - Figure 4.4.

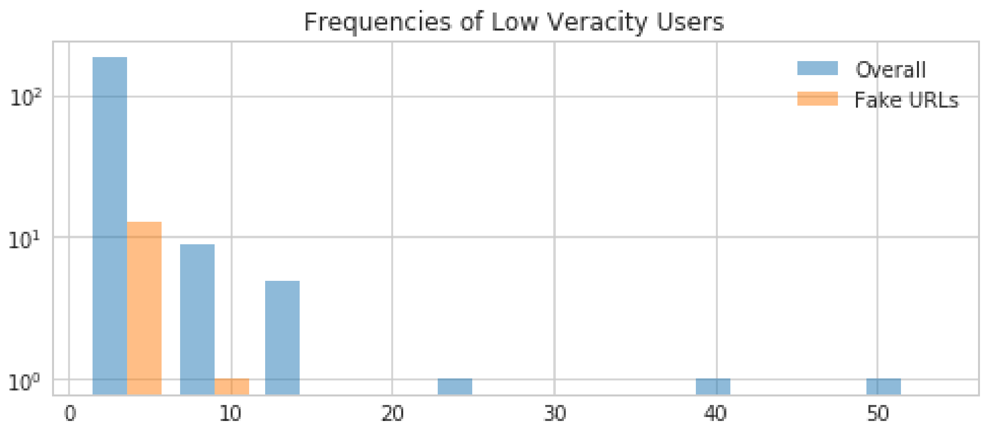


FIGURE 4.3: Frequency of low belief bucket

Furthermore, as we move on to the higher-belief groups (high - Figure 4.5 and ultra high 4.6), users show a significant increment on their overall activity as well as on their suspicious.

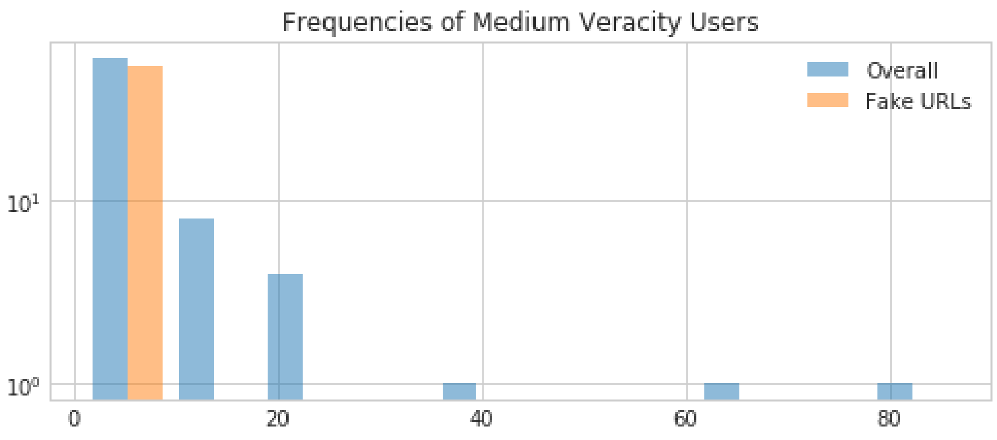


FIGURE 4.4: Frequency of medium belief bucket

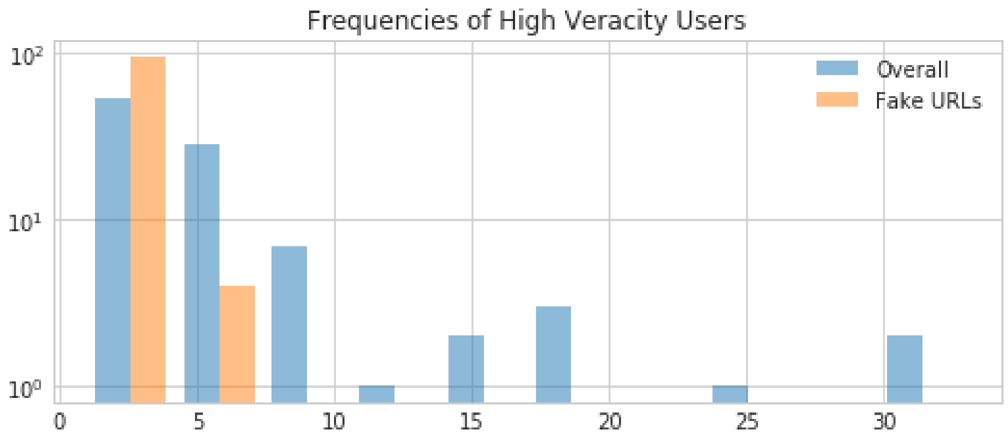


FIGURE 4.5: Frequency of high belief bucket

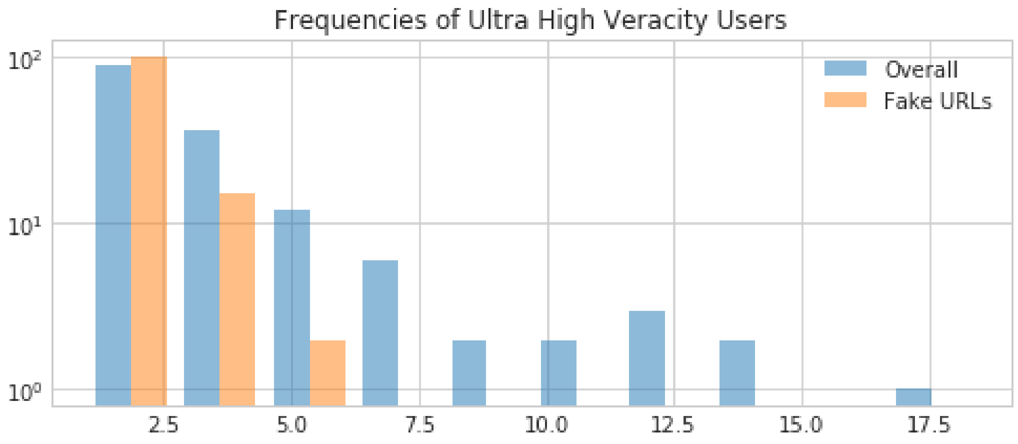


FIGURE 4.6: Frequency of ultra-high belief bucket

Chapter 5

Conclusion

Contents

5.1 Future Work

Our research results and conclusions can be effectively used in future work. As discussed in the Contributions our user blacklist will be used to construct a plugin that will be fed with a Machine Learning model that along with Natural Language processing perspective of fake news, will raise a signal that will be able to detect a fake news article-tweet as shown in Figure 5.1. Check-It's plugin is a fake news detection system,developed as a web browser plugin. Check-it aims to take a bold-step towards detecting and reducing the spread of misinformationon the Web. To do so, it empowers its users with the tools they need to identify fake news. The major challenge of fake news detection stems from newly emerged news on which existing approachesonly showed unsatisfactory performance. In order to address this issue, it proposes a pipeline based on a variety of signals, ranging from domain name flag-lists to deep learning approaches.

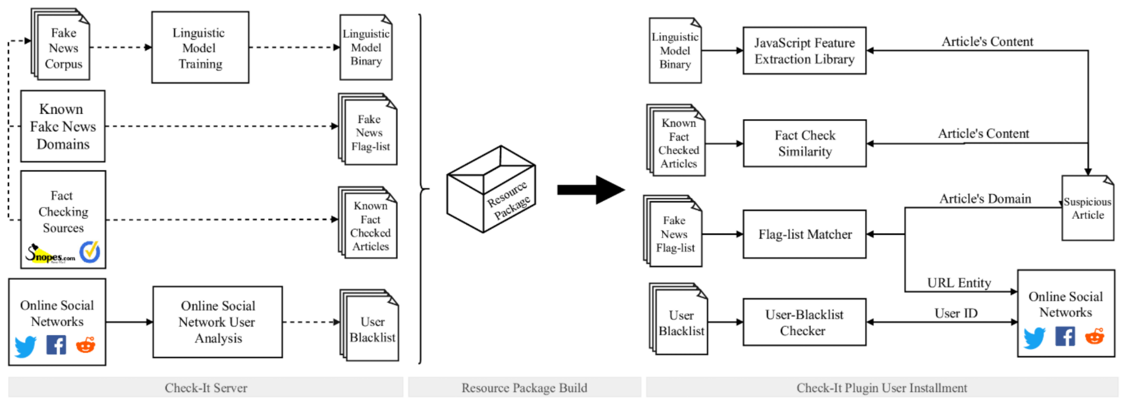


Figure 5.1: Architectural diagram for the Check-It System.

In addition, having extracted the most important features that differentiate fake news disseminators from trusted accounts we could also train a Machine Learning model that given the user-level, content-level and network-level features of a Twitter account will be in place to predict whether the user is disseminating fake news or not.

Bibliography

[1] Hunt Allcott and Matthew Gentzkow. Social Media and Fake News in the 2016 Election. *Journal of Economic Perspectives*, 31(2):211–236, may 2017. ISSN 0895-3309. doi: 10.1257/jep.31.2.211. URL http://pubs.aeaweb.org/doi/10.1257/jep.31.2.211.

[2] Soroush Vosoughi, Deb Roy, and Sinan Aral. The spread of true and false news online. Technical report.

[3] Potsane Mohale and Wai Sze Leung. Extrapolation of Aspects of Fake News on Social Networks Ubiquitous Products and Services Innovation View project Detection of Fake News on Social Networks View project Extrapolation of Aspects of Fake News on Social Networks. Technical report, 2018. URL https://www.researchgate.net/publication/326586153.

[4] Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. Mean Birds. pages 13–22. Association for Computing Machinery (ACM), jul 2017. doi: 10.1145/3091478.3091487.

[5] Manoel Horta Ribeiro, Pedro H. Calais, Yuri A. Santos, Virgílio A. F. Almeida, and Wagner Meira. "Like Sheep Among Wolves": Characterizing Hateful Users on Twitter. dec 2017. URL http://arxiv.org/abs/1801.00317.

[6] Amol Bansod. Efficient Big Data Analysis with Apache Spark in HDFS. *International Journal of Engineering and Advanced Technology*, (6):2249–8958, 2015.

[7] Morris H DeGroot. Degroot Consensus.Pdf, 1974.

[8] By Benjamin Golub and Matthew O Jackson. Naïve Learning in Social Networks and the. *American Economic Journal: Microeconomics*, 2(1):112–149, 2010.

[9] C. Romero and H. P. Oliveira. Exact solutions in brans-dicke theory: A dynamical system approach. *Astrophysics and Space Science*, 159(1):1–9, 1989. ISSN 0004640X. doi: 10.1007/BF00640482.

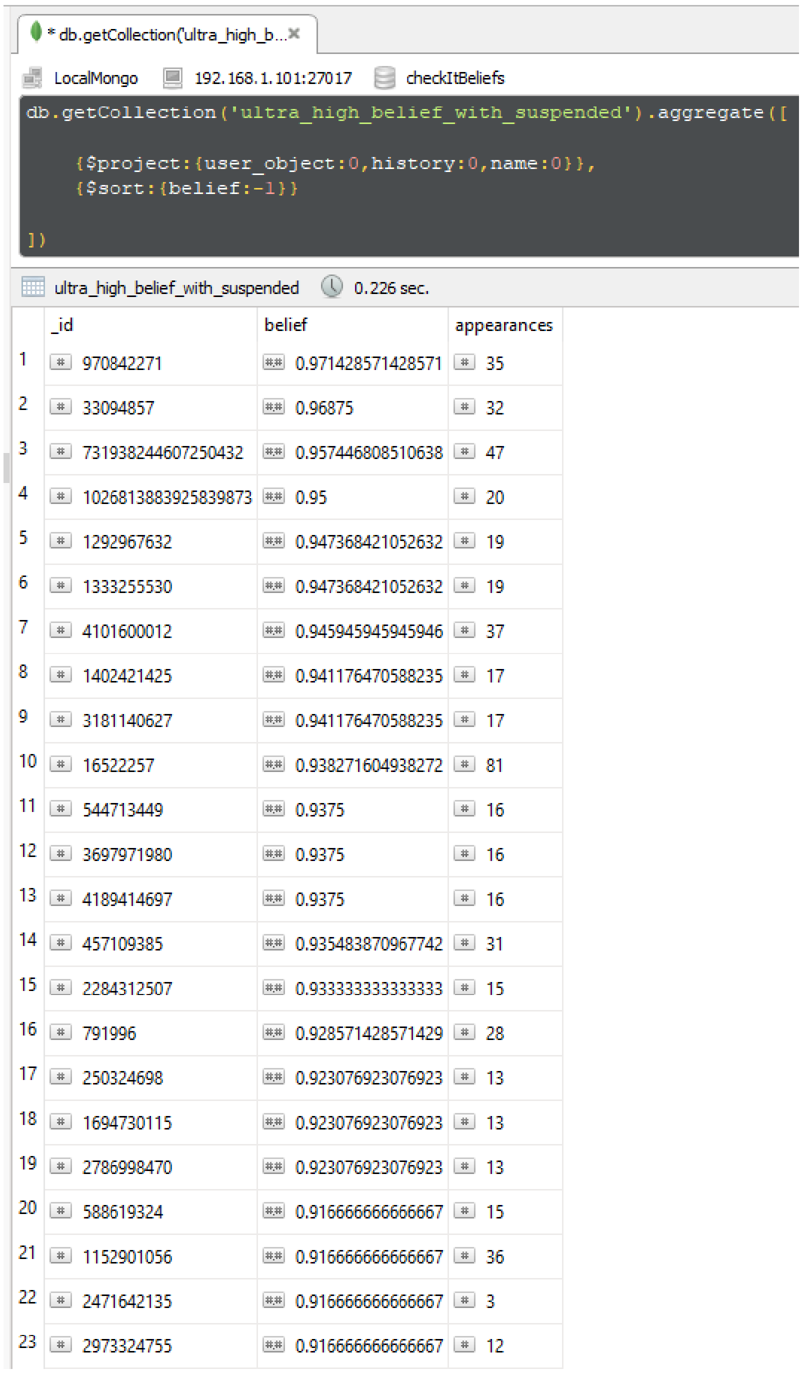
[10] Yimin Chen, Niall J. Conroy, and Victoria L. Rubin. Misleading Online Content. In *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection - WMDD '15*, pages 15–19, New York, New York, USA, 2015. ACM Press. ISBN 9781450339872. doi: 10.1145/2823465.2823467. URL http://dl.acm.org/citation.cfm?doid=2823465.2823467.

[11] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Castillo et al - Information credibility on twitter.pdf. pages 675–684, 2011. doi: 10.1145/1963405.1963500.

Appendices

Appendix A

Top 45 found Fake News disseminators



* db.getCollection('ultra_high_b...
LocalMongo
192.168.1.101:27017
checkItBeliefs

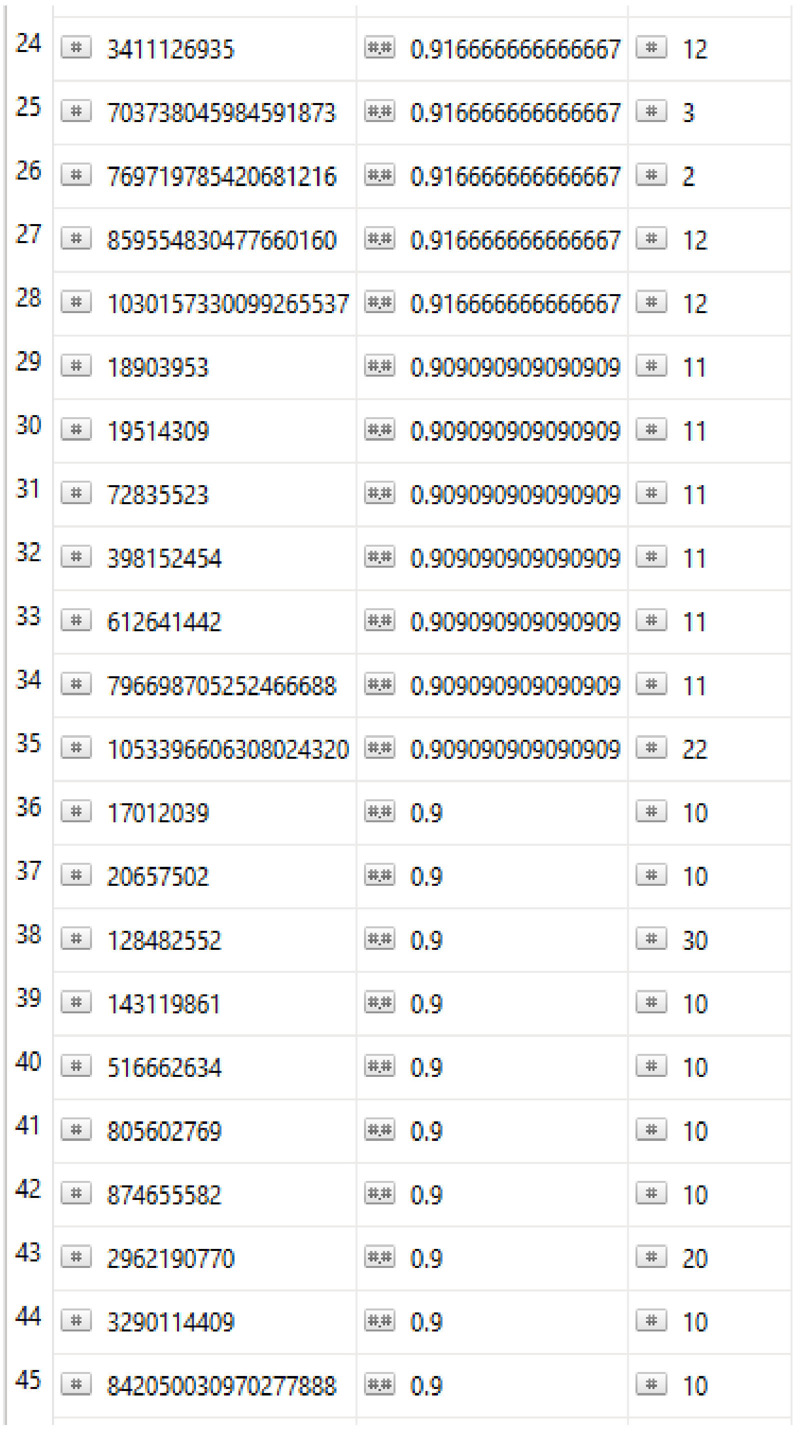
```
db.getCollection('ultra_high_belief_with_suspended').aggregate([

    {$project:{user_object:0,history:0,name:0}},
    {$sort:{belief:-1}}

])
```

ultra_high_belief_with_suspended
0.226 sec.

	_id	belief	appearances
1	970842271	0.971428571428571	35
2	33094857	0.96875	32
3	731938244607250432	0.957446808510638	47
4	1026813883925839873	0.95	20
5	1292967632	0.947368421052632	19
6	1333255530	0.947368421052632	19
7	4101600012	0.945945945945946	37
8	1402421425	0.941176470588235	17
9	3181140627	0.941176470588235	17
10	16522257	0.938271604938272	81
11	544713449	0.9375	16
12	3697971980	0.9375	16
13	4189414697	0.9375	16
14	457109385	0.935483870967742	31
15	2284312507	0.933333333333333	15
16	791996	0.928571428571429	28
17	250324698	0.923076923076923	13
18	1694730115	0.923076923076923	13
19	2786998470	0.923076923076923	13
20	588619324	0.916666666666667	15
21	1152901056	0.916666666666667	36
22	2471642135	0.916666666666667	3
23	2973324755	0.916666666666667	12



24	3411126935	0.916666666666667	12
25	703738045984591873	0.916666666666667	3
26	769719785420681216	0.916666666666667	2
27	859554830477660160	0.916666666666667	12
28	1030157330099265537	0.916666666666667	12
29	18903953	0.909090909090909	11
30	19514309	0.909090909090909	11
31	72835523	0.909090909090909	11
32	398152454	0.909090909090909	11
33	612641442	0.909090909090909	11
34	796698705252466688	0.909090909090909	11
35	1053396606308024320	0.909090909090909	22
36	17012039	0.9	10
37	20657502	0.9	10
38	128482552	0.9	30
39	143119861	0.9	10
40	516662634	0.9	10
41	805602769	0.9	10
42	874655582	0.9	10
43	2962190770	0.9	20
44	3290114409	0.9	10
45	842050030970277888	0.9	10