Dr.Watson

Project 5 - Overcoming the «fake news»

Digital Business Innovation Lab - A.Y. 2017/18
Group: *GestIT - massimo.perini@asp-poli.it*

Alessio Baccelli 898514
Gioele Bigini 898741
Filippo Calzavara 898526

Luca Comoretto 806906
Harilal Orunkara Poyil 898231
Massimo Perini 898302
Hichame Yessou 898275

Contents

Abstract

1.1 Introduction

Nowadays people retrieve pieces of information from several sources: websites, newspapers, but also social media. The purpose of this project is to explain whether the latter is a risky policy. We'll soon see why yes and we will propose an approach to tackle the issue.

Social media sites like Facebook and Twitter have become predominant tools for online users to share contents that go from simple text to rich media. The rate at which stories are getting propagated through social media is enormous. For example, on an average, around 6000 tweets are posted on Twitter per second which corresponds to 500 million tweets per day [Sta] and this number grows in an incremental fashion day by day. In recent years social media evolved as significant tool for opinion formation. It's often noticed that many incidents and stories appears in social media well before they hit on mainstream media. It played a major role in the anti-government demonstrations in Turkey [Hut] to US presidential election, 2016 [Kap]. It helps in humanitarian and disaster relief efforts and other law and order situations to a good extent. It also employs key role in the field of marketing, healthcare, politics, entertainment, etc. At the same time, the misuse of social media also gets observed frequently since several individuals are using it for promoting controversial, if not simply populist and false topics, such as anti-nationalism, human trafficking or terrorism.

To tackle the problem we first need a definition of it: fake news are false stories that appear to be news, spread on the internet or using other media, usually created to influence political views or as a joke.
More extensively, we can define them as fake news when an inaccurate, sometimes sensationalist report is created to gain attention, mislead, deceive or damage a reputation. Unlike misinformation, which is inaccurate because a reporter has confused facts, fake news is created with the intent to manipulate someone or something. Fake news can spread quickly when it provides disinformation that is aligned with the audience's point of view because such content is not likely to be questioned or discounted.

We can certainly say that social networks are very suitable to spread news seen the features of the formers: they are free and they rapidly spread information around the world, much faster than more traditional means of communication. Furthermore, they are not checked and balanced by any reliable authority. People are therefore passively fed with uncertified news which might be of several types: poorly written and therefore

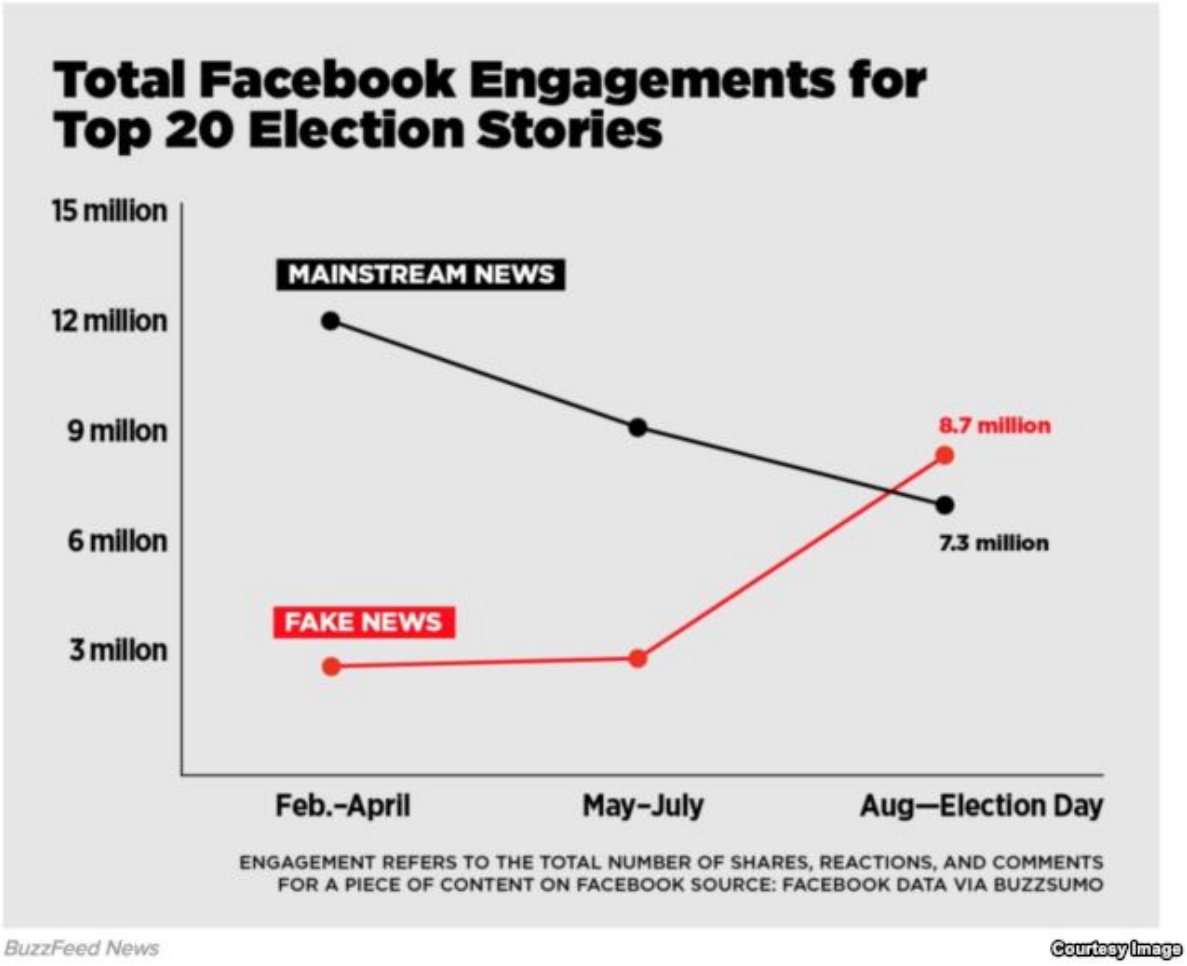


Figure 1.1: facebook engagements over news

confusing, striking and easy to contradict or misleading and made to induce them to subtly believe in something that actually hasn't happened.
This extensive spread of false reports has a deeply negative impact on the society and even single individuals, who can be led to overreact impulsively, determining dramatical and dangerous situations. There are, for example, reported episodes indeed of people shooting somebody else due to fake news Considering all these factors, a fake news detector looking after social media has recently become an important tool to distinguish and guide people in the forest of the lies. Several research groups and companies are nowadays addressing this issue investing considerable resources in order to deliver an automatized software to the market. The challenges are multiple due to the unique features that the problem presents and which make traditional algorithms rather ineffective. On one hand, the posts are written to mislead readers, recalling articles patterns so that content analysis results difficult: to address this issue is necessary to analyze further information on the person who published the news such as his relations on social media. On the other hand, the usage of this auxiliary information is not trivial at all, since they are huge and noisy: determining whether they are useful or not becomes very tricky. Additional issues are related to the human behavior: when you listen to somebody talking it's possible to understand, most of the times, the use of sarcasm or irony, in general, the tone of the speech. When it comes to reading something, all these information are lost. Wrapping up, our software has the goal to face all these issues and solve them using an ensemble of Data Mining techniques, capable of overcoming the problems.

1.2 Application Examples

1.2.1 US Election

1.2.2

Management Analysis

2.1 General Analysis

le aziente avevano il potere di controllare le informazioni che aveva il grande pubblico (comunicazione) social media -> è cambiato: strutture aziendali (internet: azienda inizia portarsi sul web) -> le persone hanno avuto possibiltà di scelta > -> ponderano meglio le decisioni sui prodotti social network + piattaforme recensione: roba ingigantita: le aziende non controllano più le informazioni che escono
prodotto -> servizio: l'azienda vuole dare una certa immagine. Le aziende hanno pensato di avere una vetrina su queste piattaforme per tenersi in contatto più facilmente coi clienti (porta molta più complessità)
non sempre il contatto diretto con gli utenti può essere positivo o negativo (se è soddisfatto o no) -> commento negativo -> pesa molto (per il social vige che se qualcuno è scontento perdi molti più clienti -> l'azienda cerca di avere sempre più clienti soddisfatti -> se è contento continuerà a comprare il prodotto)
fake news: non controllabili ma danneggiano in maniera critica l'azienda (danno di immagine) olio di palma: come per gli altri aumenta prob. tumore -> amplifica aspetto realtà con consumatori più sensibili (cancerogeno) -> fa leva su paura consumatore (citare ricerca Poli danni immagine -> danni borsa)

2.1.1 From Fake Reviews to Fake News

Opinions such as online reviews are the main source of information for customers to help gain insight into the products they are planning to buy. Customers write reviews to provide feedback by sharing their experience, either bad or good, with others. Their experiences impact businesses for the long term either positively or negatively. Naturally, this creates incentives and opportunities for manipulating customers' decisions by generating false/fake reviews. Such a practice is called opinion spamming where spammers will write false opinions to influence others. Opinions/Reviews may be produced either to improve or damage the reputation of a business/product. Recently, it has become apparent that opinion spam does not only exist in product reviews and customer feedback. In fact, fake news and misleading articles is another type of opinion spam.
One of the biggest sources of spreading fake news or rumors is, of course, social media websites,11 such as Google Plus, Facebook, Twitter, and so on. Like fake reviews, fake news can also be categorized broadly into 3 groups. The first is false news, which is news that is completely fake and is made up by the writers of the articles. The second

group is fake satire news, which is fake news whose main purpose is to provide humor to the readers. The third group is poorly written news articles, which have a certain degree of real news but are not entirely accurate. In short, it is news that uses, for example, quotes from political figures to report an entirely fake story. Usually, this kind of news is designed to promote a certain agenda or biased opinion.
Detecting fake news is believed to be a complex task and much harder than detecting fake product reviews. The open nature of the web and social media, in addition to the recent advance in computer technologies, simplifies the process of creating and spreading fake news. While it is easier to understand and trace the intention and the impact of fake reviews, the intention and the impact of creating propaganda by spreading fake news cannot be measured or understood easily. For instance, it is clear that fake review affects the product owner, customers, and online stores; on the other hand, it is not easy to identify the entities affected by the fake news. This is because identifying these entities requires measuring the news propagation, which has shown to be complex and resource intensive. [ATS17]

2.2 Current state of the Fake News issue

2.2.1 Patent research

We conducted an online research of the trend in the patents related to fake news topics [Per].

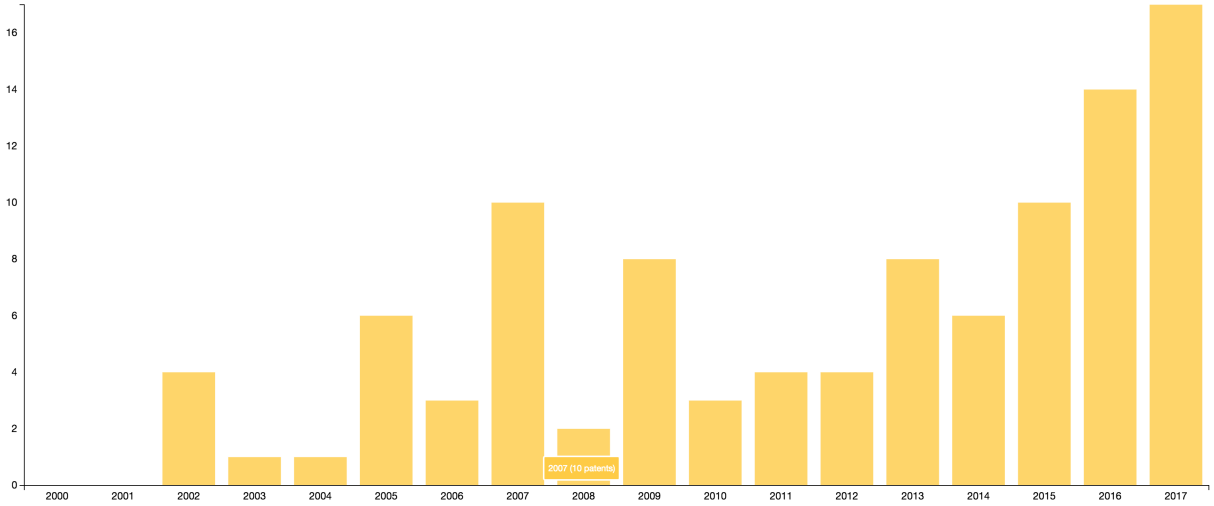
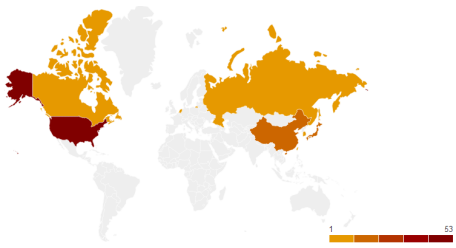


Figure 2.1: Patents related to fake text in different pubblication years



(a) Applicants of patents of each country

(b) Top 10 applicants

Brand Identity

3.1 Brand Identity and Design

An important aspect of the project development concerned the design of a Brand Identity that characterized in an unique fashion the final product delivered. The inspirational colors and shapes of the whole front-end and presentation stack resembles the undeniable suggestion of confidence that IBM first-rate services offer. The choice to include a Brand Identity chapter in this document stems from the fact that, due to the high complexity of the evaluation algorithm, a good visualization mask needed to be developed in order to convey the robustness and sophistication that the algorithmic part present, while enabling not experts of the Machine Learning field to understand the potential of this tool.
Certainly, a future possible development is the production (thus public distribution) of the platform as example of combined AI technologies applied to the real life.

3.2 Colors shades

Figure 3.1: Palette of colors used on the front-end/presentation layer of the project

The first choice that the UX unit has taken was to generate a palette of colors on which would be later drawn the template and the presentation layout.
As described before, the original colors were taken from IBM Corporate Identity and readapted for the project purposes

Color	Name	HTML HEX	RGB
	Daintree	#022231	2,34,49
	Green Vogue	#022b48	2,43,72
	Tarawera	#0c3d5f	12,61,96
	Chathams Blue	#0e486f	14,72,111
	Blue Chill	#0d6797	13,103,151

3.3 Brand creation

Inspired by the name of IBM's utilized service "Watson", we appeal to our project the name "Dr.Watson", a fictional character in the Sherlock Holmes stories by Sir Arthur Conan Doyle, while *GestIT* remains the group formal name.
The brand creation begun with logo's designing. Such process included a through research of the main font, later found as *Merriweather Black* and the application of the Palette colors.

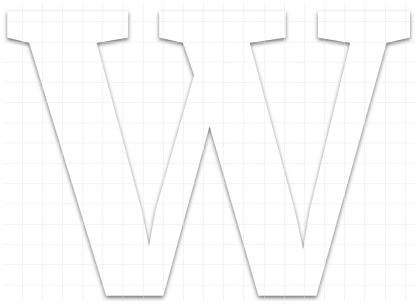


Figure 3.2: First sketch of "W" which stands for Watson of "Dr.Watson"

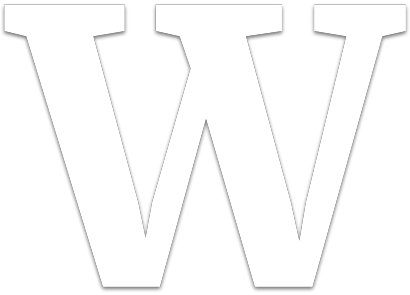


Figure 3.3: Final "W" with transparent background

Figure 3.4: Final "W" with dark blue background

Front End

The Front-end is the application layer usually referred to the client of a Client-Server architecture. Its goal is to facilitate the end user interaction with the system, composed of algorithms, through a more concrete and closer to reality interface.

4.1 Functionalities

Dr. Watson is a web application that allows the users to check and visualize statistics about tweets that may or may not contain fake news. To do so, it has to be an easy-to-use web application, with only a few steps to assess the tweet and the possibility to have the same user experience across desktop and mobile browsers. The application is able to classify a sample of test data which the user can choose from, getting a confidence interval of the belonging to the class with the prevalence of the sentiment detected by Dr. Watson. For transparency, we added some sections to have a general overview of how Dr. Watson works and the deliverables needed for the project.
We split the web application into six main sections that help the final user assess and understand the fake news issue.

4.2 Website Organization

Dr. Watson has been designed on a singular web page in order to require the lowest possible number of steps from the user and avoid long wizards that could distort the user experience on the web application.

4.2.1 Sidebar

From the left side a sidebar is available. Here It is possible to shift quickly to the preferred section available. The six horizontal lines below the section names are lighted up when selected. If no selection is made, the current section is lighted up by default.

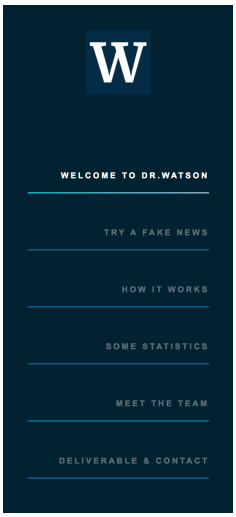


Figure 4.1: The sidebar of Dr. Watson

4.2.2 Sections

Landing Page: This is the main section when the user open the wep page. From this section the user will be scrolled down to the second section, where he is allowed to try the fact checker.

Try Fake News: This section allows checking Fake News by simply clicking on the sample shown. This will check the veracity of the tweet of will show on the right the outcome of the analysis carried out by Dr. Watson. In order to keep a high simplicity across different devices, we chose to show only three tweets per time, while allowing the user to shuffle across the dataset of tweets.

How it works: Hereby we are showing a high-level overview of how the system is structured. As stated on the website, Dr. Watson is a complex application that has been built with several programming languages. The front end interface is built in Node.js that is a JavaScript runtime built on the JavaScript Engine V8 of Chrome. The web-scraper used to retrieve, build and enrich the dataset has been written in Java interfacing with the Twitter APIs. The Machine Learning model has been written in Python in order to exploit libraries like Pandas where we have expertise.

Statistics: Visualization is a big part of the Data Science field, allowing either the developer and the user to have a look at mistakes, biases and other unexpected problems related to data. The growing availability of data has led to an increasing reliance on the data visualization providing a powerful way to communicate data-driven findings, motivating analyses and detecting flaws. In our case, we have a massive amount of data so we needed a trivial way to explain the distribution of the data and its analysis. We opted for 4 graphs, 3 of which describes the best the tweets, news and users in our dataset. The sentiment graph instead shows the distribution of the feelings contained in the tweets.

Meet the team: This section describe the whole team and development organization

that we are glad to cite again:

- Alessio Baccelli - Artificial Intelligence development
- Gioele Bigini - Front-end and Data Visualization
- Filippo Calzavara - User Experience and Front-end development
- Luca Comoretto - Artificial Intelligence developer
- Harilal Orunkara Poyil - Artificial Intelligence and Domain Expert
- Massimo Perini - Artificial Intelligence development
- Hichame Yessou - Front-end and Data Visualization

Deliverable & Contact: These final section has been used to deliver the midterm documents required for mid-term meetings with professors.

4.3 Development

4.3.1 Strategy

The chosen development strategy for Front-end development is the Top-Down strategy. The Top-Down approach generally involves identifying the final product and then solving the sub-problems deriving from it, in a divide-et-impera style.
Down below the followed procedure for the project development is summarized:

1. Generation of the HTML structure starting from an open-source template
2. Welcome page and footer development
3. Contact section development
4. Test section development
5. Statistics section development

The design of the web page has been developed concurrently to the other tasks.

4.3.2 Programming Lenguages

The three main technologies used for web development are:

- **HTML**
- **CSS**
- **Javascript**

HTML (Hyper Text Markup Language) is the standard markup language for creating Web pages. It is mainly used to build the general structure of the website.

CSS (Cascading Style Sheets) is a style sheet language used for describing the presentation of a document written in a markup language like HTML. Briefly, It is used to display contents correctly. The nature of HTML does not allow to describe the design of the contents like layouts, colors and fonts. The HTML and CSS combination constitutes the User Interface (UI), that part that can improve content accessibility, provide more flexibility and control.

Javascript (JS) enables interactive web pages for the correct interaction between the end user and website. Although HTML provides basic functions for interactions with the end user, Javascript extends the potential of HTML and CSS through animations and interactions in real time. This part constitutes the User Experience (UX) that tries to improve the overall experience of a user using the product.

4.3.3 Design Pattern

There are different design pattern to use when creating a new project:

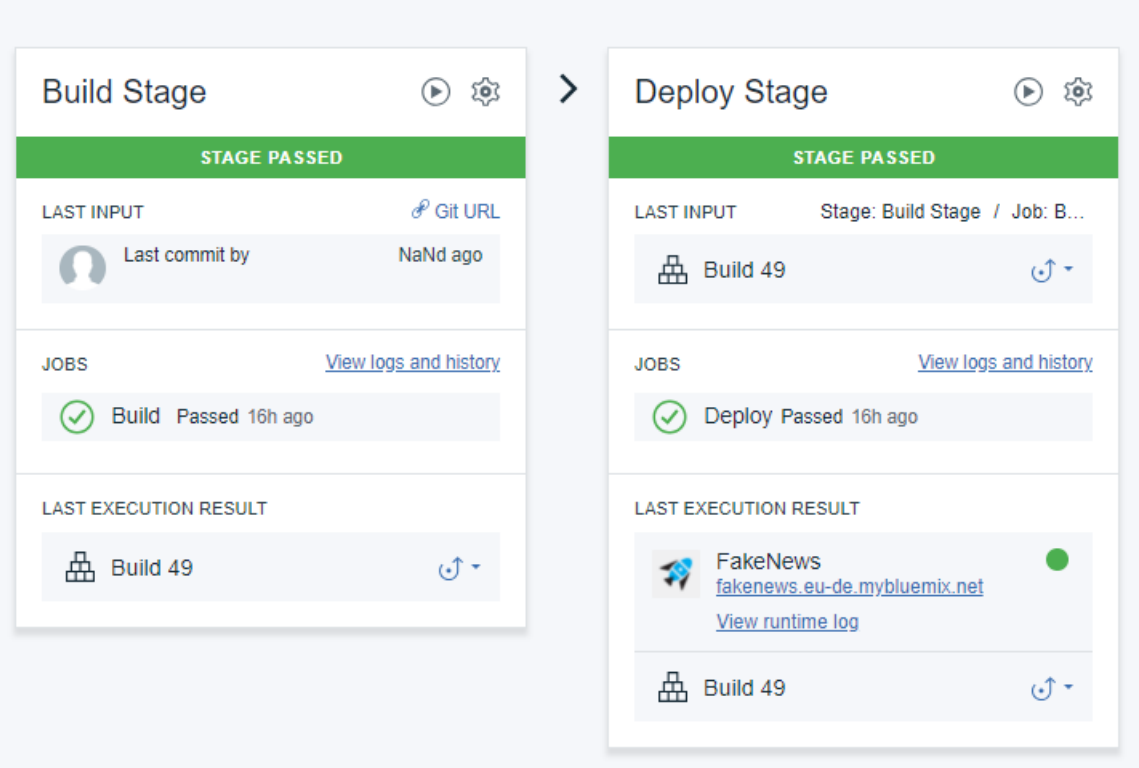
- MVC Model-View-Controller: Our Application server will use Express, a "fast, unopinionated, minimalist web framework for Node" that allows to write easily a MVC application. AngularJS is a MVC framework too.
- Proxy (Middleware): Express' Router-level middleware allows us to delegate specific handlers for each URL. In that situation it's really helpful because it provides a very clean separation between what the client "sees" and the actual logic behind it
- Observer: This is one of the key-pattern of NodeJs. Observer is a solution for modeling the reactive and asynchronous nature of Node.js (together with callbacks).
- Module pattern: The Module pattern is used to mimic the concept of classes (since JavaScript doesnt natively support classes) so that we can store both public and private methods and variables inside a single object. That allows us to create a public facing API for the methods that we want to expose to the world, while still encapsulating private variables and methods in a closure scope.
- Client-Server: The application is based on a Client-Server communication model. The clients are thin, in order to let the application run on low-resources devices and in order to easily add new features without always require an update of the application. This approach has been chosen for different reasons:
  - Its pratical and easy to implement.
  - Data synchronization: there is only a central server that keeps and manage the data.
  - Having a cluster of servers improves the availability, reliability, safety and maintenability of our system.

– Improves the security between clients, that known only the server endpoint but not other clients.

Other commonly adopted design patterns includes the Factory pattern and the Decorator pattern.

4.3.4 Automatic Deployment

In projects where several engineers work together is a crucial point to have a centralized codebase and a unique deployment point in order to be coherent and updated with the features integrated. Since structurally the project is developed in two different applications we considered necessary to split the work in different repositories while leaving a link between them. This has been done setting up the main repository which has a submodule towards the front end project in order to have only commits concerning the web application. The second part has been accomplished by having an automatic deployment system that runs after every commit in the repository. We used the IBM Delivery Pipeline in order to trigger the build process and the deployment of an instance of the Node.js Cloud Foundry applications. This provides us with the possibility to commit a new functionality and within minutes to have it deployed on the domain of our web application with the advantage of testing it on several desktops and smartphones while keeping a high level of integrity.



4.3.5 Tools

Visualizing dataset insights is not that easy in general. A powerful tool to do it has been Highcharts, a Javascript library that easily allow developers to set up interactive charts

in a web page. It is fully HTML5 compatible and dynamic that means hardly customizable.

4.4 Code structure

Our front-end has been developed with Node.js and Express in order to have a minimal and robust framework that we could have shaped with our needs and integrated with the several APIs which Dr. Watson needs. We chose to use Handlebars as a template engine since we wanted to keep the majority of the business logic away from the representational layer of our web application. Every component and section of Dr. Watson has been build in order to be completely responsive and to be correctly displayed on the majority of the display sizes.

The web application is structured around the landing page, which allows us to have a very simple routing structure, with very few routes used for interactions with Twitter APIs. This was needed in order to allow the integration of multiple features from several engineers at different times. Having a strong and neat separation of concerns have been a key factor in the early decisions regarding the architecture of the web application and the structure of work-flow.

The entry point of our application is the `/bin/www` that initialize the low-level settings like port usage, HTTP server and the various listeners that it needs. Our core component is the app.js which set up the routes and the templating engine of our web application. From here we render all the sections of the landing page through the index route, while the tweet route is left to interact with the Twitter APIs. We mainly have used 2 Twitter APIs which are the `statuses/show/:id` and the `statuses/oembed` that have been used to retrieve the JSON data of the tweets and the HTML attribute in order to embed the Tweets.

4.4.1 Fake News Evaluation

Figure 4.2: The icon indicates that a Fake News has been found

The network speed is an important bottleneck to keep in count. Since the computation of the models have been done through the IBM Watson API we opted for an Offline-based computation where few results have been pre-computed and available to be shown to the end-user. To evaluate if a news is Fake or Real the user can easily use the tool available inside the "Try a Fake News" clicking on the preferred tweet. The Offline-computation will start showing the results.

4.4.2 Statistics

We had several technologies to choose from to visualize our data, but we opted for Highcharts because it seemed to be the most responsive and with the highest number of features to be personalized.

The Javascript code of the Highcharts implementation in the public folder and they are fed with JSON, originating from the same folder, in order to enhance and speed up the loading time.

The JSON files have been created in the pre-processing phase. This phase has been useful to correctly reshape data without incurring in problems such invalid values or outliers. Furthermore, since the whole dataset is too large, the pre-processing phase permitted to focus over those data useful to represent the charts in every specific case.

Four are the visualization created, corresponding to four different statistics:

- Dataset Insights

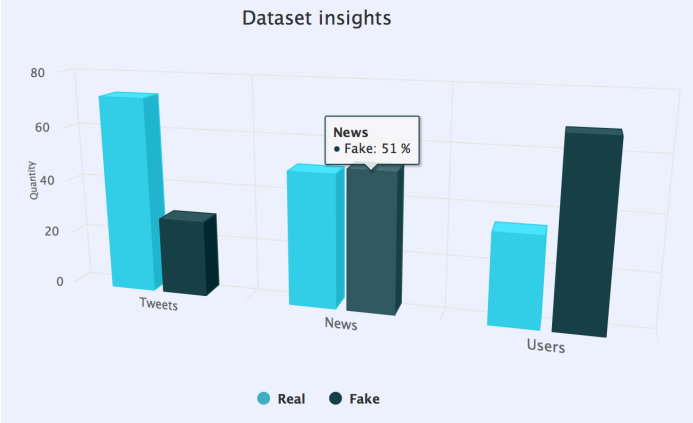


Figure 4.3: The histogram with the main features of a tweet

Sometimes tweets are relating to the same news. The chart above assures that the analyzed data have been equally distributed and what emerges is that generally the fake news are more popular with a more wide diffusion between users than real news. The reason could lie to the fact that those news tries to hit the sentiments of a user, causing him to share it again.

- Distribution of the Tweets over time

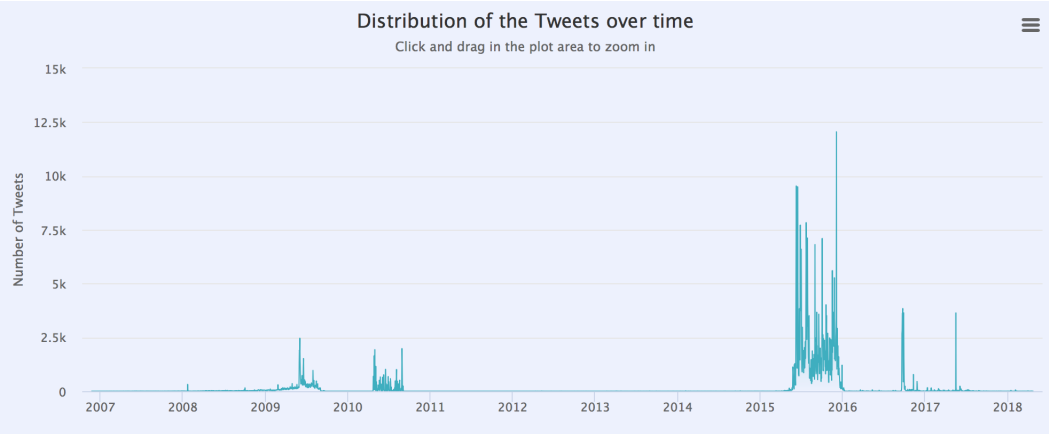


Figure 4.4: The time band considered

The representation shows the time band considered in the project. As we can see, the majority of the tweets comes from 2015 to 2017.

- Tweets Spread over the week:

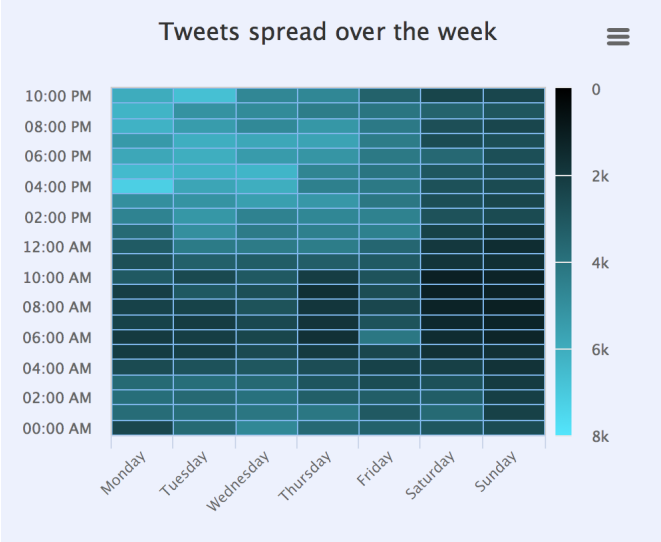


Figure 4.5: The heat map representing news spread

The heat map is really helpful to show 3-Dimension data. The color from "black" to "light-blue" represent the quantity of tweets in that specific period of time. The more near to black is the color of a cell, the less tweets have been created and obviously the more near to light-blue is the color of a cell, the more tweets have been created.

What emerges from the plot is that seems the majority of the news are created on Monday and Tuesday evening.

- Sentiment Distribution over Tweets:

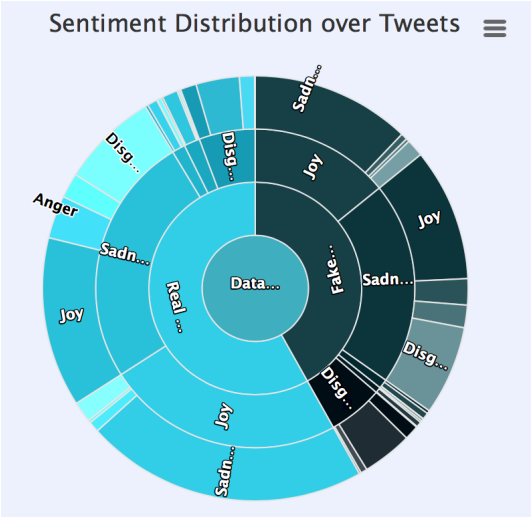


Figure 4.6: The sunburst chart representing sentiment over tweets

The plot show the relation between sentiments and tweets that could be real or fake. The interesting chart shows that the majority of the fake news are trying to hit two sentiments: Joy or Sadness. Actually, these sentiments seems to lead users to share again the news.

Fake News Detection using Social Media

5.1 Introduction

Detecting fake news on social media is an important but also a challenging problem. There are several difficulties related to this task: one of these is that even the human eye cannot accurately distinguish true from false news. For example, one study found that 80% of high school students had a hard time determining whether an article was fake or not [Edk16]. Another one is the highly dynamical nature of social media: each seconds more than 8000 tweets are written: we should have a fast way to detect the spreading fake news. In this section, a model able to detect fake news using only data related to social media (in our case Twitter) will be outlined. Models that use the article texts will probably be more accurate than this one, but the one here explained will definitely be faster since relies only to data owned by social network companies.
This model relies on the response that a news article would generate in the users: how a certain article makes them feel? Efforts to automate response detection typically model the spread of news in the social graph or use social-network dependent features. For example, [TBD+17] shows that the number and frequency of Facebook likes can be a predictor of a post with a fake article. Unfortunately, access to a social graph is not feasible in practice and manual selection of features is labor intensive.

5.2 Data collection

For this project, we make use of two publicly available datasets: the one used in [MGM+16] paper and [Shu17], which contain the set of tweet ids for different fake news articles. We crawled corresponding tweet contents, using Twitter streaming API [Twi18]. We then extracted the URLs mentioned in each tweets and crawled the text content of the URLs. For this we used Jsoup [Hed10], a Java based library for HTML parsing. From the response JSONs obtained using crawling phase we filtered the fields used in analysis. The fields we have identified for analysis and the corresponding descriptions are given in 5.1. The overall statistics of the final dataset is given in 5.2.

5.2.1 Training data for analyses

We created 4 different datasets for building analysis models. These datasets are different based on the number of articles(crawled using the links specified in the tweets) per event. The distributions are 5, 10, 15 and 20 articles per event.

Table 5.1: Descriptions of data fields

Field	Description
id	The integer representation of the unique identifier for one Tweet
tweet_content	The actual UTF-8 text of the status update
url_content	The content(if available) of any URL mentioned
event_id	The unique identifier for the event(fake news) this tweet belongs to
created_at	UTC time when this Tweet was created. Example: "created_at":"Wed Aug 27 13:08:45 +0000 2008"
user_id	The integer representation of the unique identifier for the the user who posted this Tweet
in_reply_to_status	If the represented Tweet is a reply, this field will contain the integer representation of the original Tweet's ID. Otherwise represented using NULL string
hash_tags	Hash tags, which have been parsed out of the text of the Tweet
user_mentions	Mention of other Twitter users, which have been parsed out of the text of the Tweet
retweet_count	Number of times this Tweet has been retweeted
favourite_count	Indicates approximately how many times this Tweet has been liked by Twitter users
possibly_sensitive	This field only surfaces when a Tweet contains a link. The meaning of the field doesn't pertain to the Tweet content itself, but instead it is an indicator that the URL contained in the Tweet may contain content or media identified as sensitive content
place_name	When present, indicates that the tweet is associated (but not necessarily originating from) a Place
place_type	Type of place(admin, country, poi, city, neighborhood) as given by Twitter API
country_code	The 2 letter code representation of country from which the tweet orginated
coordinates	The (approximate, not necessarily) the tweet get tweeted
is_fake	1 and 0 for the fake and real respectively

Table 5.2: Dataset Statistics

Total number if rows(contain tweets for both real and fake news)	580898
Number of fake events	1296
Number of tweets representing real news	416319
Number of tweets representing fake news	164579
Number of unique users	237878
Number of tweets which are shared ones(retweets)	16003
Number of tweets contain hash tags	99068
Number of unique hash tags	22130
Number of tweets contain user mentions	119790
Number of tweets with more than one retweet count	63473
Number of tweets with more than one favorite count	60242
Number of tweets marked as possibly sensitive by Twitter	12875
Number of tweets with location information	5931
Number of unique places	1800
Number of unique place types	5
Unique place types	admin, country, poi, city, neighborhood
Number of unique countries from which tweets originated	98
Top countries of origin of tweets with tweet count	US:4654, GB:449, CA:162, NG:75, IN:52, AU:52, DE:37, ZA:32, IE:29, MX:28, BR:22, FR:21, KE:15, PH:15, NL:14, DK:12, JM:11, IL:10, IT:10, GH:10, PK:10
Number of tweets with URL mentions	306836

5.3 Model description

We consider a series of temporal interactions that occurred between n users with m news-articles over time [1,T]. Each engagement between a user u_i and an article a_j at time t is represented as $e_{ijt} = (u_i, a_j, t)$. This model adopt three main features:

- Textual information: features related to the text of the interaction
- Response of users: features related to frequency and distribution of the interactions
- Source: features that represent the author of the interaction

5.3.1 Response features

In order to fully track the response to an article with high efficiency, we generate different partitions p for each event. Each partition groups interactions according their normalized creation time (the first interaction will be at time 0) and their number is a hyperparameter of this model. For each partition we have several features. The first two are η and Δt. η is the number of interaction in a partition, Δt is the time between the first interaction of the partition and the last interaction of the last not-empty partition. The usage of Δt allows to remove empty partitions since they don't have anymore useful informations. The partitions are then padded in order to have the same number of partitions for each article.

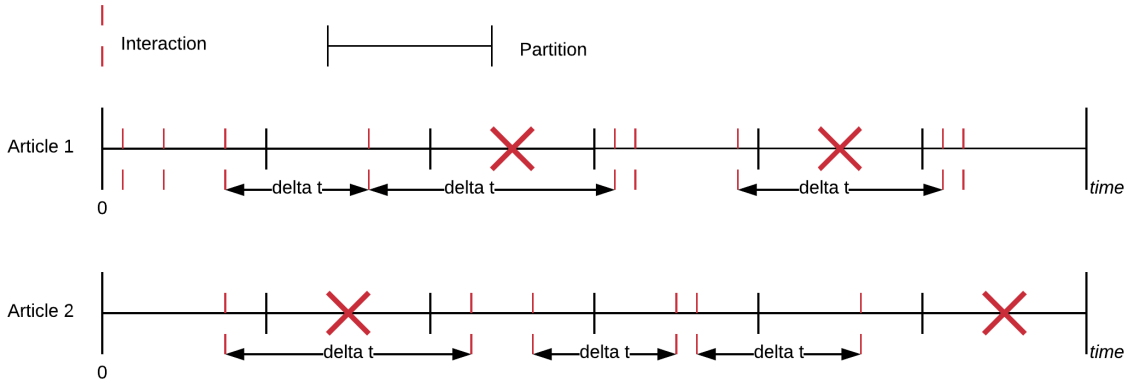


Figure 5.1: Response features

5.3.2 Text features

Text feature vector is related to the text available in each interaction with a given article a_j. The first step has been the generation of clean texts, so we removed the punctuation symbols from the original interactions. Then we choose the Natural Language Processing doc2vec [LM14] Neural Network model to avoid hand-crafted textual feature selection. In this way we automatically extracted a compressed and less noisy representation of the text of each interaction. Hyperparameters of doc2vec will therefore be hyperparameters of our model. EXPLAIN PARAMETERS HERE

5.3.3 Source features

Source feature vector is a vector for each user and not a different vector for each given article. We want represent users by the articles each one interacted with. To do so, in line with existing literature on information retrieval and recommender systems, we construct the binary incidence matrix of which articles a user engaged with. This matrix is a (n, m) matrix in which

$$\begin{cases} x_{i,j} = 1, & \text{if } \exists t \mid \exists e_{ijt} \\ x_{i,j} = 0, & \text{otherwise} \end{cases}$$

Since the number of 0 will be much higher than the number of 1, in order to be more efficient and avoid memory waste, we implemented it as a sparse matrix. We therefore apply the Singular Value Decomposition (SVD) to extract a lower-dimensional representation for each u_i

5.3.4 Prediction model

We can formalize some points we discussed so far: A is the set of articles, U is the set of users, T is the discrete set of time the articles have been written. P is the set of partitions and $P_{ji} \subset Pj \subset P$ is the $i-th$ partition linked to event j. e_{ijt} is an interaction of user $i \in U$ to article $j \in A$ at time $t \in T$. L is the length of each partition. F_{ijk} is the value of the $i-th$ feature of the partition $j-th$ of event k

$$\forall j \in A, \exists i \in U, \exists t \in T \mid \exists e_{ijt}$$
$$e_{ijt} \in Psu \leftrightarrow (j = s \wedge t \geq (L \cdot u) \wedge t < L \cdot (u+1))$$
$$\forall i \forall j \forall k \nexists e_{abc} \in P_{jk} \implies F_{ijk} = 0$$

From the above conditions is clear that each article can now be seen as a multivariate time serie. Each partition of each article has N different variables:

- η
- Δt
- The averages of the text features (computed over interactions in a partition)
- The averages of the source features (weighted computation over users whose interactions are in the partition)

Table 5.3: Example of multivariate time series

event	partition	η	Δt	0	1	2	3 ...
1	0	3	0.0	5.066724e-03	6.042654e-05	0.717824	0.858997
1	1	1	50643000.0	1.487121e-05	2.883997e-08	0.001058	0.000248
1	2	2	122161000.0	2.265753e-09	2.081532e-10	0.000016	0.000020
2	0	1	0.0	3.168470e+00	1.224494e-03	8.467130	2.965042
2	1	2	43371000.0	2.625489e-03	1.898865e-05	0.110333	0.020215

This table shows a sample of time series. For each article there are N partitions and for each partition there are features related to user interactions and text written (DA MIGLIORARE)

The problem now is related to the classification of multivariate time series. We implemented with Tensorflow + Keras the state-of-the-art Deep Learning MLSTM-FCN model described in [KMDH18]. The input to this Neural Network will be a 3-dimensional matrix $(events, features, partitions)$ and the output will be binary. This model comprise of a fully convolutional block and a LSTM block, as shown in 5.2. The fully convolutional block contains three temporal convolutional blocks, used as a feature extractor. Each convolutional block contains a convolutional layer, with filter size of 128 or 256, and is succeeded by a batch normalization. The batch normalization layer is succeeded by the ReLU activation. In addition, the first two convolutional blocks conclude with a squeeze and excite block. The final temporal convolutional block is followed by a global average pooling layer.

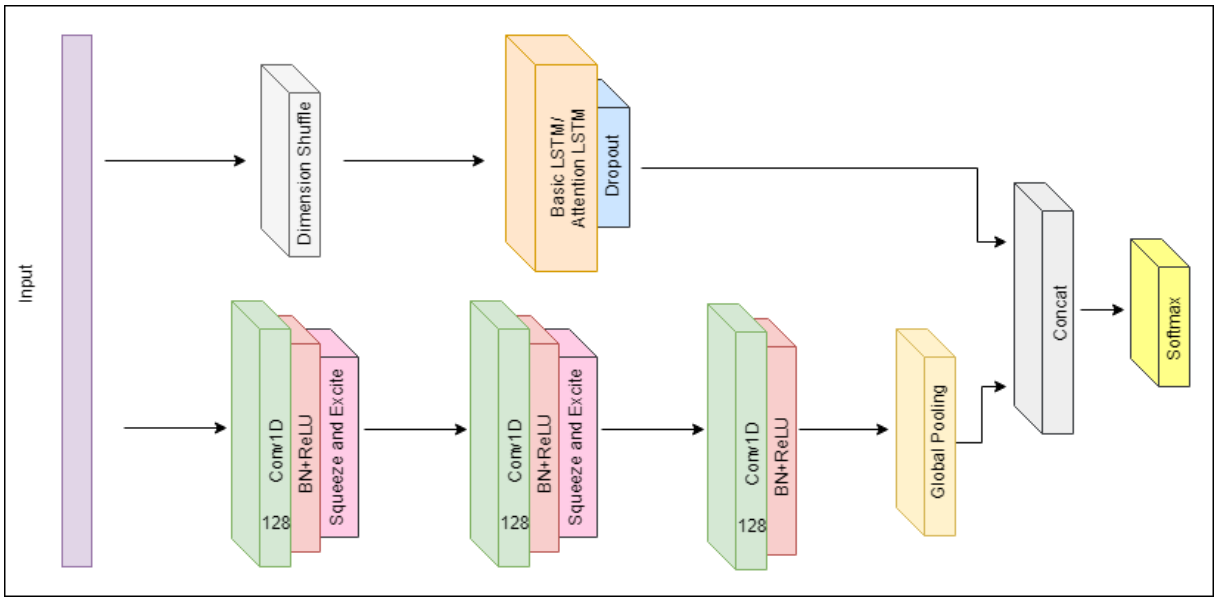


Figure 5.2: The MLSTM-FCN architecture [KMDH18]. LSTM cells can be replaced by Attention LSTM cells to construct the MALSTM-FCN architecture [KMDH18]

5.4 Results

We evaluated this model with K-Fold Cross Validation on 10 folds. Average accuracy ...

Back End

6.0.1 FastText

FastText is a library for learning of word embeddings and sentence classification created by Facebook's AI Research (FAIR) lab. The model is an unsupervised learning algorithm for obtaining vector representations for words [JGBM16]. The confusion matrices of FastText models for different datasets(different number of articles per event) are given in 6.1. The average accuracy of these models is 68.56956%.

Table 6.1: Confusion matrices of FastText models

Number of test samples : 1248 (Training data contain 5 posts per event) Accuracy : 70.03205%			Number of test samples : 2275 (Training data contain 10 posts per event) Accuracy : 71.82417%		
	Fake(Predicted)	Real(Predicted)		Fake(Predicted)	Real(Predicted)
Fake(Actual)	320	197	Fake(Actual)	605	309
Real(Actual)	172	559	Real(Actual)	326	1035
Number of test samples : 3242 (Training data contain 15 posts per event) Accuracy : 69.95682%			Number of test samples : 3951 (Training data contain 20 posts per event) Accuracy : 62.4652%		
	Fake(Actual)	Real(Predicted)		Fake(Predicted)	Real(Predicted)
Fake(Actual)	1053	523	Fake(Actual)	419	1318
Real(Actual)	441	1225	Real(Actual)	157	2057

Sentiment Analysis

7.1 Data collection and Preprocessing

SEE WHAT TO WRITE AFTER THE MERGE OF THE INTRODUCTION FROM HARI AND MASSIMO

7.2 IBM Watson NLU

The IBM Watson Natural Language Understanding (NLU) is a really powerful service, provided by IBM in cloud, that allows the user to analyze a sentence, a paragraph or even an entire article. The functionalities provided by this service are many:

- Sentiment analysis
- Emotion analysis
- Identify keywords, entities, concepts and their relations

After trying different combinations and settings, we've decided to analyze the texts of the articles linked to every tweet with IBM Watson NLU in order to extract keywords and the attributes associated to each of them. The outputs of the NLU for each keyword are: the score for the general sentiment, the score for each emotion (joy, sadness, disgust, fear, anger) and the score for the relevance of the keyword in the sentence.

7.3 Text extraction, data cleaning and features

As mentioned above, the analysis was done over the articles contained in the tweets. Specifically, the text of the articles was retrieved by using an existing API [Hed10]. After the analysis performed by the NLU, we've cleaned the data by removing all the tweets which had a "Null" score in the output of the sentiment and of the emotional analysis. Indeed, the URL contained in these kind of tweets were leading to non-articles pages, like links to YouTube videos and to other tweets and therefore could not be analyzed by the NLU. Moreover, since the project was considering only English tweets and articles, all the non-English text was not analyzed and the corresponding tweet was removed from the dataset.
As features, to compute a single value for each emotion and for the general sentiment, we've decided to perform an average of the scores of the target emotion (or sentiment)

of all the keywords identified by the NLU, weighted with the relevance in the phrase of the keyword itself.

7.4 Training and testing

For computational issues and because of the limit to the request that can be done to an IBM Watson API, the model was trained and tested with a small fraction of the total dataset. In particular, the dataset was divided by event and 80% of it was taken for training and the remaining part for testing. Moreover, because of the reasons mentioned above, only 5 tweets per event were taken to build the new dataset.
After different tests and evaluation, the method with the best performance over the dataset was a Quadratic Discriminant Analysis (QDA), with an accuracy of 66% on the test set.

Table 7.1: Confusion matrix for QDA using 5 tweets per event

	True (Predicted)	False (Predicted)
True (actual)	520	77
Fake (actual)	252	118

Bibliography

[ATS17] H. Ahmed, I. Traore, and S. Saad, "Detecting opinion spams and fake news using text classification", vol. 1, e9, Dec. 2017.

[Edk16] B. Edkins. (2016). Americans believe they can detect fake news. studies show they can't, [Online]. Available: https://www.forbes.com/sites/brettedkins/2016/12/20/americans-believe-they-can-detect-fake-news-studies-show-they-cant/#5798f89e4022 (visited on 05/20/2018).

[Hed10] J. Hedley. (2010). Jsoup java html parser, [Online]. Available: https://jsoup.org/ (visited on 04/15/2018).

[Hut] S. Hutchinson. (). Social media plays major role in turkey protests, [Online]. Available: http://www.bbc.com/news/world-europe-22772352 (visited on 06/02/2018).

[JGBM16] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification", Jul. 2016.

[Kap] M. Kapko. (). How social media is shaping the 2016 presidential election, [Online]. Available: http://www.cio.com/article/3125120/social-networking/how-social-media-is-shaping-the-2016-presidential-election.html (visited on 06/02/2018).

[KMDH18] F. Karim, S. Majumdar, H. Darabi, and S. Harford, *Multivariate lstm-fcns for time series classification*, 2018. eprint: arXiv:1801.04503.

[LM14] Q. Le and T. Mikolov, "Distributed representations of sentences and documents", in *Proceedings of the 31st International Conference on Machine Learning*, E. P. Xing and T. Jebara, Eds., ser. Proceedings of Machine Learning Research, vol. 32, Bejing, China: PMLR, 22–24 Jun 2014, pp. 1188–1196.

[MGM+16] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha. (2016). Detecting rumors from microblogs with recurrent neural networks, [Online]. Available: http://alt.qcri.org/~wgao/data (visited on 05/01/2018).

[Per] M. Perini. (), [Online]. Available: http://www.patentinspiration.com (visited on 06/02/2018).

[Shu17] K. Shu. (2017). Kaidmml dataset, [Online]. Available: https://github.com/KaiDMML/FakeNewsNet (visited on 04/17/2018).

[Sta] I. L. Stats, Ed. (), [Online]. Available: http://www.internetlivestats.com/twitter-statistics (visited on 06/02/2018).

[TBD+17] E. Tacchini, G. Ballarin, M. Della Vedova, S. Moret, and L. de Alfaro, "Some like it hoax: Automated fake news detection in social networks", Sep. 2017.

[Twi18] Twitter. (2018). Consuming streaming data, [Online]. Available: `https://developer.twitter.com/en/docs/tutorials/consuming-streaming-data.html`.