Natural language processing for scam detection. Classic and alternative analysis techniques

Author: Ignacio Palacio Marín
Supervisor: David Arroyo Guareño

A thesis submitted in fulfillment of the requirements for the degree of
Msc in BigData and DataScience

September 28, 2019
“Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism.”

Dave Barry
UNIVERSIDAD AUTÓNOMA DE MADRID

Abstract

Escuela Politécnica Superior

Msc in BigData and DataScience

Natural language processing for scam detection. Classic and alternative analysis techniques

by Ignacio PALACIO MARÍN

We have seen, over the past decades, an overwhelming increase in the volumes of information being generated, distributed and shared, specially but not only, though social media networks. It has led to an exponential growth on the importance of data in the decision taking processes of most industries and economic sectors, proving the criticality of ensuring the quality of the information we are gathering and using.

Whilst most of this information is, or at least is intended to be, true, a non-negligible portion of it contains false information. Miss-information campaigns have played an important role in recent and critical decision-taking processes such as the Brexit referendum or the 2016 U.S. presidential elections.

The current spread of incorrect information constitutes a meaningful potential risk on information systems’ management. This problem becomes even greater when considering decision taking automatic algorithms. As a matter of fact, social media and opened access to data may constitute a way to break the information’s asymmetry that has traditionally affected areas such as the financial industry.

This paper will propose different techniques of natural language processing, from the more traditional ones to a brief approach to more recently developed techniques on deep learning approaches. They are all intended to enable an automatic texts’ classification in different discussion forums and constructing procedures to pursue users or groups of users’ classifications as an open gate to generate attribution procedures for information sources.
Acknowledgements

I would first like to thank my thesis advisor David Arroyo Guardeño. His office door was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right direction whenever he thought I needed it.

Finally, I am pleased to express my very profound gratitude to my wife and loving friend, Luisa-Pilar, for her unfailing support and continuous coverage through my years of study and the process of researching and writing this thesis. This accomplishment would not have been possible without her. Thank you.

Ignacio Palacio Marín
Contents

Abstract v
Acknowledgements vii

1 Introduction 1
  1.1 Motivation ................................................. 1
    1.1.1 False information taxonomy ......................... 2
    1.1.2 Regulatory concern ................................. 3
  1.2 Objectives ............................................. 4
  1.3 Thesis organization .................................... 5

2 Strategies for campaigns against fake news 7
  2.1 State of the art .......................................... 7
    2.1.1 Following the path of flawed information: network analysis and hub characterization .......................... 7
    2.1.2 Content based verification ........................... 9
    2.1.3 Detection using computational methods ................ 10
  2.2 Our proposal’s approach ................................ 13
    2.2.1 Text structure ...................................... 13
    2.2.2 What? How? ......................................... 15

3 Proposed techniques: Conventional NLP approach 17
  3.1 As a consequence of the “no free lunch” theorem ....... 17
  3.2 Methodology ............................................. 17
    3.2.1 Token based analysis ............................... 21
    3.2.2 Embedding based analysis ........................... 22
  3.3 Testing ................................................ 23

4 Proposed techniques: Advanced deep learning techniques 27
  4.1 Methodology ............................................. 27
    4.1.1 Long Short-Term Memory (LSTM) neural networks ...... 28
    4.1.2 Gated Recurrent Units (GRU) neural networks .......... 29
  4.2 Testing ................................................ 30

5 Conclusions and future work 33
  5.1 Conclusions .............................................. 33
  5.2 Future work ............................................ 34

A Topic modelling visualization 35
B Implemented algorithm architecture diagram 37
C Classification algorithms parametrization 39
D LSTM and GRU networks underlying equations
  D.1 LSTM hidden layer equations 43
  D.2 GRU equations 43

Bibliography 45
List of Figures

1.1 Fake news - Google Trends ................................. 1
1.2 Bloomberg - Dow Jones industrial average index ............ 3

2.1 Polifact - Barack Obama’s file ................................ 9
2.2 Twitter dataset - Stories content distribution ................ 14
2.3 Two labeled Twitter dataset - Ordered countplot ............ 14
2.4 Two labeled Twitter dataset - Stories content distribution . 14
2.5 Cloud representation - Stories lemmatization content representation . 15

3.1 scikit-learn - Machine learning algorithms cheat sheet ........ 18
3.2 Traditional NLP techniques - Accuracy and roc_auc scores .... 25

4.2 Greff et al. - LSTM unit ....................................... 28
4.3 LSTM vs GRU units ............................................ 30
4.4 Twitter stories length distribution .............................. 30
4.5 LSTM: Loss/Accuracy scores - sigmoid activation function .... 31
4.6 LSTM: Loss/Accuracy scores - linear activation function .... 31
4.7 GRU: Loss/Accuracy scores ................................... 32

5.1 Obtained accuracy and roc_auc scores ......................... 33
5.2 Obtained precision, recall and F1 scores ...................... 34

A.1 Coherence score - Topic classification .......................... 35
A.2 pyLDAvis - Topic visualization ................................. 36

B.1 Text classification proposed algorithms architecture .......... 37
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Zannettou et al. [84] - On the detection of false information</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>Yang et al. [79] - Test set accuracy results by step</td>
<td>12</td>
</tr>
<tr>
<td>3.1</td>
<td>Random Forest - Grid searched parameters</td>
<td>19</td>
</tr>
<tr>
<td>3.2</td>
<td>k-NN - Grid searched parameters</td>
<td>19</td>
</tr>
<tr>
<td>3.3</td>
<td>SVM - Grid searched parameters</td>
<td>20</td>
</tr>
<tr>
<td>3.4</td>
<td>XGBoost - Grid searched parameters</td>
<td>20</td>
</tr>
<tr>
<td>3.5</td>
<td>Traditional NLP - ML accuracy and roc_auc scores</td>
<td>24</td>
</tr>
<tr>
<td>4.1</td>
<td>LSTM - Grid searched parameters</td>
<td>31</td>
</tr>
<tr>
<td>C.1</td>
<td>TF-IDF - Classification algorithms parameterization</td>
<td>39</td>
</tr>
<tr>
<td>C.2</td>
<td>Word2Vec - Classification algorithms parameterization</td>
<td>40</td>
</tr>
<tr>
<td>C.3</td>
<td>Doc2Vec - Classification algorithms parameterization</td>
<td>41</td>
</tr>
</tbody>
</table>
List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLP</td>
<td>Natural Language Processing</td>
</tr>
<tr>
<td>NLI</td>
<td>Natural Language Inference</td>
</tr>
<tr>
<td>IR</td>
<td>Information Retrieval</td>
</tr>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>AI HLEG</td>
<td>Artificial Intelligence High-Level Expert Group</td>
</tr>
<tr>
<td>CCN-CERT</td>
<td>Centro Criptológico Nacional - Computer Emergency Response Team</td>
</tr>
<tr>
<td>HBPT</td>
<td>Homogeneity-Based Transmissive Process</td>
</tr>
<tr>
<td>TLV</td>
<td>Tweet Latent Vector</td>
</tr>
<tr>
<td>OSN</td>
<td>Online Social Networks</td>
</tr>
<tr>
<td>FNC</td>
<td>Fake News Challenge</td>
</tr>
<tr>
<td>TF-IDF</td>
<td>Term Frequency – Inverse Document Frequency</td>
</tr>
<tr>
<td>LD</td>
<td>Linear Discriminant</td>
</tr>
<tr>
<td>RF</td>
<td>Random Forest</td>
</tr>
<tr>
<td>SVM</td>
<td>Support Vector Machine</td>
</tr>
<tr>
<td>k-NN</td>
<td>k-Nearest Neighbours</td>
</tr>
<tr>
<td>XGBoost</td>
<td>Extreme Gradient Boosting</td>
</tr>
<tr>
<td>ROC</td>
<td>Receiver Operating Characteristic</td>
</tr>
<tr>
<td>AUC</td>
<td>Area Under Curve</td>
</tr>
<tr>
<td>TPR</td>
<td>True Positive Rate</td>
</tr>
<tr>
<td>FPR</td>
<td>False Positive Rate</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi Layer Perceptron</td>
</tr>
<tr>
<td>RNN</td>
<td>Recurrent Neural Network</td>
</tr>
<tr>
<td>CNN</td>
<td>Convolutional Neural Network</td>
</tr>
<tr>
<td>LSTM</td>
<td>Long Short-Term Memory</td>
</tr>
<tr>
<td>SA-LSTM</td>
<td>Sequence Autoencoder Long Short-Term Memory</td>
</tr>
<tr>
<td>C-LSTM</td>
<td>Contextual LSTM</td>
</tr>
<tr>
<td>HAN</td>
<td>Hierarchical Attention Network</td>
</tr>
<tr>
<td>Conv-HAN</td>
<td>Convolutional Hierarchical Attention Network</td>
</tr>
<tr>
<td>BoW</td>
<td>Bag of Words</td>
</tr>
<tr>
<td>GRU</td>
<td>Gated Recurrent Units</td>
</tr>
<tr>
<td>BERT</td>
<td>Bidirectional Encoder Representations (from) Transformers</td>
</tr>
<tr>
<td>LDA</td>
<td>Latent Dirichlet Allocation</td>
</tr>
<tr>
<td>LSA</td>
<td>Latent Semantic Analysis</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation

Currently data is a critical fuel to the prevailing economic and social systems. It has definitively become a commonplace to refer it as the ‘new oil’ of the global economy [18]. Therefore, techniques and infrastructure to store, process and analyse it are as well in the core of most of the latest technology trends.

Decision making processes, at governmental, enterprise and even personal levels, are increasingly sustained on a permanently growing amount of data, leading, on an accelerating pace, to a data-driven world.

Organizations of all kind now have stockpots of raw data along with sophisticated analysis tools that combined drive their decision-making processes when trying to create new market opportunities. Decisions are no longer made based on gut instinct but on pure evidence, or so they expect to be. When analysing the evolution of companies’ investments and strategies over the past decade big bets on data and analytics are quite common and almost obliged.

The increasing concern over the recent years as well as the importance of fake news is easily understood when addressing Google trends, see Figure 1.1, to evidence how fake news has gained popularity worldwide.

![Figure 1.1: Fake news. Source: Google Trends](image)

The wide spread of fake news may have a negative impact on society and individuals. An increasing exposure, and therefore desensitization, to fake news definitively affects the way people approach and respond to real news and undermine their capability to separate what is true from what is not. It is important to understand that even though fake and captious news have existed from long ago we are witnesses to an overwhelming increase tied up with the increase of information we are exposed to.
Furthermore, there is overall consensus that good data is important for any analysis but there is no independent definition of what constitutes good data and what does not \[25\].

### 1.1.1 False information taxonomy

A clear description of every faced problem is always solid ground over which start discussing and proposing new approaches. Fake news won’t be different in that matter.

Zanettou et al. proposed, on their study of false information on online social networks [84], a taxonomy of the Web’s false information ecosystem, based on [73] and extended. A brief description of this classification is detailed bellow.

Based on the types of false information Zanettou et al. distinguished between:

1. **Fake news**, identifying the following types of false information within it:
   - *Fabricated*, or completely fictional stories.
   - *Propaganda*, aiming to hurt the interest of a particular party or promote those from another.
   - *Imposter*, news whose author or source has been impersonated.
   - *Conspiracy theories*, addressing an event or set of circumstances invoking a conspiracy without further evidence.

2. **Biased/Inaccurate news**, as to news that are misleading but do reflect truth in some extent:
   - *Hoaxes*, containing false or inaccurate facts being presented as legitimate.
   - *Hyper-partisan*, referring to stories that are extremely one-sided.
   - *Fallacy*. News making invalid reasoning when constructing an argument.

3. **Misleading/Ambiguous news**, including:
   - *Rumors*, widely disseminated news with no discernible source and without known authority for its truth.
   - *Click-bait*, or the deliberate use of misleading headlines.
   - *Satire news*, where its’ main concern is the irony or humor touch they are draw up with, usually obfuscating the facts.

They also elaborate on the taxonomy of false information actors, identifying a handful of different players such as *bots*, *criminal/terrorist organizations*, *political persons*, *hidden paid posters and state-sponsored trolls*, *journalists*, *trolls and useful idiots*. Being the latest those users who share fake news because they are manipulated or because they are naive.

Besides, they proposed as well an assortment of the underlying motives for the propagation of false information, including, *malicious intent, political influence, profit, passion and fun*. 
1.1. Motivation

1.1.2 Regulatory concern

The outstanding rising of false news is not only a problem of misinformation but it has become an increasing concern for a growing number of actors and may be considered as one of the greatest threats to journalism and, what is even more worrisome, to democracy itself [88]. Examples of such are the 2016 U.S. presidential election campaign, where top twenty fake election stories generated above 8M interactions on Facebook [86][5][28], and the 2016 Brexit referendum, where the Digital, Culture, Media and Sport Committee concluded that "the UK is clearly vulnerable to covert digital influence campaigns" [34].

Furthermore, even though misinformation and financial markets has been a long run problem for a while, the raising of the fake news phenomenon has taken it to a new level. In 2013, over $130 billion, see Figure 1.2, were wiped out after a ‘bogus’ tweet suggesting that Barack Obama had been injured in an explosion [57].

![Figure 1.2: Source: Bloomberg - Dow Jones industrial average index, April 23rd 2013 plunge.](image)

As a matter of fact, the erosion to democracy, and public trust posed by the unprecedented growth in misinformation has led to a sustained demand of further analysis of this phenomenon [87], as to newer regulation at all levels [34].

Moreover, legislation recently sanctioned, in both Germany and France [34], come up as meaningful examples of local regulation against harmful online content. Both examples constitute successful demonstrations as to how local Governments can enforce rules empowering judges to order the immediate removal of disinformation online articles. Paraphrasing the French formula, both regulations seek “information that is fair, clear and transparent” on personal data usage sanctioning actions like “deliberately disseminate false information likely to affect the sincerity of the ballot”.

Earlier this year, the European Commission High-Level Expert Group on Artificial Intelligence (AI HLEG) delivered the Policy and investment recommendations for trustworthy Artificial Intelligence [1], establishing as a point of special criticality the

---

1 Lower house of the Parliament of the United Kingdom.
study of the relationship between AI and Cybersecurity to develop and support AI-specific cybersecurity infrastructures to leverage Europe’s enablers for trustworthy AI. It does state that "user centric cybersecurity policies must protect users’ rights online, by design and by default”.

Additionally, the AI HLEG dedicates a full chapter to “Establishing an appropriate governance and regulatory framework” encouraging the definition of an appropriate framework, ensuring and respecting fundamental rights, in order to safeguard individuals and society. Finally, among the fostered measures presented in their conclusions they promote the "data-driven economy as a cornerstone for the EU’s future success in the global market”.

Finally, local Spanish authorities such as Centro Criptológico Nacional (CCN-CERT), recently published as well a reference guide [6] to help addressing the main features and methodology of the current disinformation techniques convening citizens and digital media users to acquire the necessary tools and skills so they can consume and share information avoiding unwillingly spreading miss-information. Recently, attacks to a country’s interest have migrated from targeting their IT infrastructure or communications systems, corporate or governmental, into trying to undermine an essential gear of modern democracy, public opinion.

As a matter of fact, according to the CCN-CERT guide [6], there are, at least, six factors contributing to the increasing frequency of miss-information campaigns:

- High level of effectiveness, along with limited costs and broad diffusion.
- Difficulty establishing a direct attribution of the miss-information campaigns. Anonymous profiling, automation messages distribution processes, software and technology helping to cover trace and IP addresses, etc. highly limit the tracing of the information and its sources as well as the reaction capability to establish a direct attribution of the attacks.
- Complex regulation as in opposition to classical opened battlefield war.
- Causal relationship establishment limitations.
- Usage of existing social vulnerabilities.
- Infiltration of illegitimate misinformation in legitimate social and political communication channels, challenging the audience capability of separating fact from fake and discern what opinions and information among those being distributed are or not legitimate.

The CCN-CERT guide suggests as well a decalogue of recommendations to "communication war victims": analyzing the information sources, being aware of distributed screenshots, analyzing who shared a certain piece of information and in what context, detecting bots or multiple-profile actors, read into the details, enhance critical thinking and conscience of the impact of individual actions, etc.

1.2 Objectives

This thesis intent is to present a broad comparison of various machine learning and deep learning techniques trying to identify fake news from a sample of texts. Further
detail will be presented if future chapters, nevertheless we depict in this section an
general overview of the developed implementations and analysed framework.

Using the already cleaned and tagged Twitter based dataset used by Jooyeon Kim et al. on their HBPT\(^2\) proposal [42], a road-map from the more traditional natural lan-
guage processing, hereinafter NLP, techniques to newer deep learning implementa-
tions such as LSTM\(^3\) neural networks is presented.

It is not within the objectives of this master thesis to provide a deeper extent or
profound analysis of any of the proposed classification algorithms, nor of the used
analysis techniques.

### 1.3 Thesis organization

This master thesis memorandum is opened into the following chapters:

- **Proposed techniques: Conventional natural language processing approach.**
  - *State of the art*, a critical review in the context of the presented work and
    related issues.
  - *Our proposal’s approach*, as to the different approaches presented on the
    State of the art, this section introduces this thesis implemented approach.
    - Text structure, an overview on the used data-set.
    - What and how, further detail on the proposed approach.

- **Proposed techniques: Advanced deep learning techniques.**
  - Methodology, presenting a description and justification of the methods and
    approaches used.
  - Testing, detail of the performed tests and obtained results.

- **Conclusions**, presenting a wider perspective including further implications of
  the obtained results and future work on the matter.

---

\(^2\) Homogeneity-Based Transmissive Process.

\(^3\) Long short-term memory.
Chapter 2

Strategies for campaigns against fake news

2.1 State of the art

As to the more recent and meaningful research made over the classification, propagation and cascade patterns, analysis or separating algorithms for fake news we will present hereafter a carefully selected picture of the existing papers on the matter that have enlightened this thesis.

Furthermore, Zannettou et al. provided a detailed overview of the most relevant research papers following their own classification proposal, on their study of false information on online social networks [84], providing great understanding on users’ perceptions on false information, its’ propagation, detection and containment, and a review the most relevant studies of fake information on the political stage.

The fake news phenomenon does include, in fact, miss-information, dis-information and mal-information, along with all their modalities [74].

2.1.1 Following the path of flawed information: network analysis and hub characterization

There is a quite a long history regarding the research on the very foundations of miss-information. There are many theories as to why the very nature of them is so rooted on our history. From the application of the early theories of Eric Fromm [21] where he explored humanity’s relationship with freedom, to the confirmation bias theory\(^1\) [52], and Barabasi’s proposal [4], considered, in fact, the very origin of network analysis studies. However, people actually are not keen on separating fact from fake, only about 70\% to 75\% of humans are actually able to effectively identify fake news [54] on a broader context [59].

The detail presented hereinafter regarding the dishonest information propagation phenomenon will be presented by means of its application to the fake news problem in hand.

Moreover, Del Vicario et al. provided evidence on how users tend to gather themselves in communities by interest causing reinforcement on their opinions and shared

\(^1\) Seeking or interpreting of evidence in ways that are partial to existing beliefs, expectations, or a hypothesis in hand.
content, and fostering confirmation bias, segregation and polarization at the expense of the quality of the shared information leading to the proliferation of biased narratives [16].

Recent work on the differences between true and false information in social networks by Vosoughi et al. based on their classification by topics, cascade patterns and sizes, diffusion speeds, emotion, and sentiment [72], leading to some meaningful conclusions including that false information is diffused significantly farther, faster, deeper, and more broadly. As to the spreading pattern of true and false news in social media networks [78], early-fact-check [68] and building fact checking systems [69] there are multiple academic papers demonstrating the significant progress made over the recent years.

On the role of user’s profiling, i.e. node and hub analysis, and its’ relationship with fake news Raymond S. Nickerson [52] presented a deep dive on its understanding, demonstrating that user’s features can provide a key stone on helping to detect fake news. When considering the profiling option on the analysis, usually both explicit and implicit features are taken into account [64].

Extended examples of explicit users’ features analysis may be credibility classification or identity linkage. In terms of credibility assessment, news-worthy topics can be separated from others by analyzing users’ explicit features such as the extension and amount of previously written messages by user [8]. As to identity linkage, users’ unique behavioral patterns and users’ information redundancies due to these behavioral patterns can be tied up together [83].

On the other hand, implicit features can be exploit to, for example, attribute age from writing style, even predicting gender or personality as in Schwartz et al. [61], where from Facebook messages and through personality tests they identified variations in language by characterizing personality, gender and age. Finally, political bias can also be deduced [11].

Comprehending the diffusion process of fake news is another key aspect to understanding the phenomenon and, obviously, social media networks have a great share on it. Information’s diffusion is in the core of the very nature of social media, however, inflammatory and emotional news spread faster than real news. Therefore, understanding their mushroom on social networks may have an impact on addressing their detection [53].

Furthermore, Shao et al. studied on [63], by means of 14 million messages spreading 400 thousand articles on Twitter during and following the 2016 U.S. presidential elections, the role that social bots play on the diffusion of fake news, providing evidence of the unequal role they play on the process. Altogether with how vulnerable we are to miss-leading information and manipulation suggests that restraining social bots may be an effective countermeasure.

Propagation based methods, for fake news detection, have gained popularity over the past years. For example, Zhiwei Jin et al. proposed a hierarchical propagation model based on user posts and considering credibility propagation through a three-layer credibility network to detect fake news [38].

Further use of propagation based methods include the application of epidemiological modeling techniques. Fang Jin et al. proposed on [37] a characterization of tweets diffusion though a comparison of different epidemiological models to characterize
2.1. State of the art

propagation cascades. Finally, statistical modeling has also been used when analyzing the propagation of fake news on online social networks.

2.1.2 Content based verification

There are as well, investigations on the linguistic side of fake news, demonstrating the positive impact of fact-checkers [36]. As a matter of fact, big platform companies such as Google or Facebook have increased their reliance on fact-checking as a response to the raising on miss-information. They have both deployed fact-checking services to help determine the veracity of factual statements in social media [12][26].

Broadly speaking fact-checking techniques may be divided into expert-based, crowd-sourced and automatic fact-checking systems [87]. While expert-based approaches rely on domain-experts, are easy managed and often lead to high-quality results. They are, however, costly and hardly scalable. Actually there are quite a number of websites allowing expert-based fact-checking. See Polifact caption on Figure 2.1.

On the other hand, crowd-sourced fact-checkers rely instead on large populations of regular individuals. This alternative is deeply connected with the concept of collective intelligence by means of the use of collaboration, collective efforts and competition of many individuals to produce a consensus decision making. They are however, when compared with expert-based systems, difficult to manage, less credible and accurate. Unlike expert-based, crowd-sourced fact-checking websites are still in early stages. This systems do rely, as a matter of fact, on the concept of “wisdom-of-the-crowds”, however ”stupidity-of-the-herds” is still a challenge they face, actually limiting their effectiveness [82].

However, traditional approaches based on human verification do not scale to the volume generated in online social networks, thus this requires the development of new computational techniques to complement expert verification.
In any case, the role of the human being in news’ fact-checking processes is still open and under discussion. Opened discussion schemes are human-in-the-loop, human-on-the-loop and human-out-of-the-loop, terms in fact used on military operation strategy [65], but increasingly applied on civil contexts.

### 2.1.3 Detection using computational methods

Automatic detection techniques have been developed to address the lack of scalability of manual fact-checking systems. Actual developments rely, mostly on Information Retrieval (IR) and Natural Language Processing (NLP) techniques, as well as on graph theory and other models/techniques.

Zannettou et al. collected and presented a well detailed and quite extensive summary of the works addressing the false information detection problem [84]. Papers presented in such table considering only Twitter and Facebook are detailed on Table 2.1 bellow.

<table>
<thead>
<tr>
<th>Social Network</th>
<th>Machine learning</th>
<th>Deep learning</th>
<th>Other algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Twitter</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Castillo et al. [8]</td>
<td>Volkova et al. [70]</td>
<td>Resnick et al. [58]</td>
<td></td>
</tr>
<tr>
<td>Gupta et al. [29]</td>
<td></td>
<td>Vosoughi et al. [71]</td>
<td></td>
</tr>
<tr>
<td>Kwon et al. [45]</td>
<td></td>
<td>Jaho et al. [35]</td>
<td></td>
</tr>
<tr>
<td>Yang et al. [80]</td>
<td></td>
<td>Qazvinian et al. [56]</td>
<td></td>
</tr>
<tr>
<td>Liu et al. [47]</td>
<td></td>
<td>Zhao et al. [85]</td>
<td></td>
</tr>
<tr>
<td>Wu et al. [77]</td>
<td></td>
<td>Farajtabar et al. [19]</td>
<td></td>
</tr>
<tr>
<td>Gupta et al. [30]</td>
<td></td>
<td>Kumar et al. [43]</td>
<td></td>
</tr>
<tr>
<td>AlRubaian et al. [2]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hamidian et al. [31]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Giasemidis et al. [24]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kwon et al. [44]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Facebook</strong></td>
<td>Tacchini et al. [67]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conti et al. [13]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 2.1: Zannettou et al. [84] - On the detection of false information**

As the brief overview on the loads of existing papers on the topic provided by the table above demonstrates, miss-information detection is not a straightforward task. There are, however, a number of metrics and approaches to the problem. A great number of the existing approaches try to use machine learning techniques when applying computational methods to the problem in hand. The difficulty of implementing effective human-on-the-loop schemes lies in the attacker’s ability to carry out counterintelligence dynamics, that is, adversary procedures and characterization/detection evasion techniques.

Hereinafter, we will provide an overview on the recent studies by walking through the references proposed by Zannettou et al.

On the referenced papers applying machine learning techniques on social network analysis for fake news detection, Castillo et al. [8] classified a number of tweets with precision and recall scores in the range from 70% to 80% by means of conventional machine learning techniques (SVM, Decision Trees, Decision Rules, and Bayes Networks).
Gupta et al. [29] adopted a supervised machine learning and relevance feedback approach prominent content based and user based features to rank tweets according to their credibility score. Kwon et al. [45] classified rumors with high precision and recall results in the range of 87 to 92% by taking to exam temporal, structural and linguistic aspects of diffusion by traditional machine learning techniques (Decision Trees, Random Forest and SVMs). Yang et al. [80] combined bursty term identification and multi-dimension sentence modeling, working along with conventional machine learning techniques (Naive Bayes, Logistic Regression and Random Forest), to automatically detect emerging hot topics for rumor identification.

Furthermore, Liu et al. [47] proposed a real time rumor debunking algorithm, using SVMs. Wu et al. [77] analyzed whether knowledge learned from historical data could potentially help identify newly emerging rumors. Gupta et al. [30], presented a semi-supervised tweets credibility scoring model using TweetCred real-time. Moreover, AlRubaian et al. introduced a multi-stage credibility framework using Naïve Bayes classifier obtaining accuracy 90.3%, 86.24% precision and 98.8% recall. Finally, on the Twitter based references, Hamidian et al. [31] adopted a supervised rumor classification employing the Tweet Latent Vector (TLV) feature, creating a 100 dimension vector representing each tweet, achieving a 97.2% precision with a SVM tree kernel model. Giasemidis et al. [24] aimed, over 80 trustworthiness measures from 72 identified rumours, to produce trustworthiness scores for classifying tweet with a maximum accuracy of 96.6%. Kwon et al. [44] studied the classification performance levels over varying time windows—from the first three days to nearly two months, providing insight into the cumulative spreading patterns of rumors over time.

As to those referred papers on Facebook data-sets based on machine learning analysis, Tacchini et al. [67] showed that Facebook post can be highly accurately classified based on the users who "liked" them, obtaining classification accuracy over 99%. Additionally, Conti et al. [13] focused on Online Social Network (OSN) structural properties of the information cascade as they are inherently difficult to be manipulated obtaining, on a highly imbalanced dataset and using a total of 28 features over three distinct classifiers (Linear Discriminant (LD), Random Forest (RF), Multi-Layer Perceptron (MLP)) classification F-score never exceeded 0.7.

Further work on machine learning approaches to fake news detection, even if not strictly applied to OSN and therefore not referred by Zannettou et al. [84], is hereafter introduced. Khan et al. [41] conducted a benchmark study assessing the performance of different machine learning algorithms (SVM\(^2\), Linear regression, decision trees, AdaBoost, Naïve Bayes and k-NN\(^3\), extending their research into neural network-based and deep learning models (CNN, LSTM, Bi-LSTM, C-LSTM\(^4\), HAN\(^5\), Conv-HAN\(^6\) and Char-level C-LSTM) as well.

On the other hand, as to the papers presenting deep learning approaches to OSN fake news detection, Volkova et al. [70] built predictive deep learning models to predict four sub-types of suspicious news: satire, hoaxes, clickbait and propaganda,

---

\(^2\) SVM stands for Support Vector Machines.  
\(^3\) k-nearest neighbours also referred as k-NN.  
\(^4\) C-LSTM or Contextual LSTM networks.  
\(^5\) HAN stands for Hierarchical Attention Network.  
\(^6\) Conv-HAN, Convolutional Hierarchical Attention Network.
Further work on deep learning approaches to fake news detection, even if not strictly applied to OSN analysis but still meaningful, is hereafter detailed. Davis et al. [15] introduced in the 2017 FNC-1\(^7\) four neural network models, two using a feed-forward architecture (Concatenated multi-layer perceptron (Concat MLP) and Bag-of-words multi-layer perceptron (BoW MLP)) and two using a recurrent architecture (Dual GRU and Bi-directional concatenated and stacked LSTM (Bi-dir LSTM)), achieving categorical test-set accuracy of 93%. Price et al. [55] focused their proposal on the exploitation of LSTM for classification of news by comparing the performance of a basic LSTM network against a sequence autoencoder LSTM network (SA-LSTM) obtaining accuracy scores on the test set of 78.74% and 70.61% respectively.

Besides, Mrowca et al. [51] approached the problem by feeding headlines into a Bi-directional LSTM and concatenate the output with headlines’ global statistical features achieving an overall score of 87.4% and a mean F1 score of 69.5%. Samir Bajaj [3] built a content based classifier, exploring several architectures (logistic regression, two-layer feedforward neural network, recurrent neural network (RNN), LSTM and GRU networks, Bi-dir LSTM, Convolutional Neural Network (CNN) with Max Pooling and Attention-Augmented Convolutional Neural Network) obtaining precision scores ranging from 0.87 (CNN with Max Pooling) to 0.97 (CNN with Max Pooling and Attention), recall scores from 0.003 (CNN with Max Pooling and Attention) to 0.79 (GRUs) and F1 scores from 0.06 (CNN with Max Pooling and Attention) to 0.84 (GRUs).

Additionally, there are quite a number of papers discussing the latest advances on text processing applied neural networks. Hereafter some examples of BERT\(^8\) implementation applied to fake news detection papers are presented. Yang et al. at [79] treated the tasks of natural language inference (NLI) training a number of NLI models including BERT achieving test set accuracy scores of 88.06% by blending predictions from first level ensembled models (using LightGBM and densely connected feed-forward network) along with BERT networks and the took those as soft pseudolabels and used them to fine-tune the pre-trained NLI models, using the output to re-feed a second level ensemble (LightGBM and a multi-layer perceptron) who’s output is connected to a fine tuned BERT with a final bleeding and taking into account the transitive relations on the used data.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensemble (1)</td>
<td>0.86741</td>
</tr>
<tr>
<td>BERT (1)</td>
<td>0.86689</td>
</tr>
<tr>
<td>Blended (1)</td>
<td>0.86963</td>
</tr>
<tr>
<td>Ensemble (2)</td>
<td>0.87990</td>
</tr>
<tr>
<td>BERT (2)</td>
<td>0.87484</td>
</tr>
<tr>
<td>Blended (2)</td>
<td>0.88019</td>
</tr>
<tr>
<td>Blended (2) + Transitivity</td>
<td>0.88063</td>
</tr>
</tbody>
</table>

**Table 2.2:** Yang et al. [79] - Test set accuracy results by step

\(^7\) (FNC-1), Fake News Challenge, is a public competition that aims to find automatic methods for detecting fake news.

\(^8\) BERT, Bidirectional Encoder Representations from Transformers neural networks.
The detailed accuracy results presented by Yang et al. (see Table 2.2) for every of the aforementioned steps are detailed in order to evidenced the shocking and compromising relationship between accuracy gain and complexity increase. Furthermore, Table 2.2, demonstrates the implementation effort and complexity increase required to gain not even a 2% in test set accuracy results.

2.2 Our proposal’s approach

This thesis presents a walk-through and comparison over different fake news detection approaches. It’s intention does not lay on introducing novel approaches in fake news detection problems but instead present a narrative and juxtaposition over some of the main existing techniques.

The implementation proposed on this thesis uses the cleaned dataset provided by Kim et al. on [42], previously used in rumor detection researches [48][49][50], taken from the sources shared on their project’s GitHub. It contains fact-checked Twitter rumors and the referenced articles within them. This thesis does not consider the data extraction step but instead relies on already cleaned data for its purposes.

Based on that source a three way path is proposed:

1. Implementation of traditional NLP techniques, more precisely TF-IDF.
2. Implementation of embedding based classification architectures. Word2Vec and Doc2Vec embedding algorithms are put to test.
3. Use of deep learning algorithms for text classification purposes.

News are previously treated applying standard NLP methods when required such as lemmatization, stop words removal, etc. And four machine learning classification algorithms (Random Forest, k-Nearest Neighbours, SVM and XGBoost) where applied on both the first two proposed paths (TF-IDF, Word2Vec and Doc2Vec).

Hereinafter we provide first, a full description of the used data (section 2.2.1), and then a detailed description of the implemented architecture (section 2.2.2).

2.2.1 Text structure

As previously stated, this thesis uses the cleaned dataset provided by Kim et al. on [42], previously used in rumor detection researches [48][49][50]. From the scratch Twitter information, Kim et al. pre-processing included the exclusion of news whose content could not be retrieved with the Twitter API, cleaning the stories cascades and confirming that there were no “circular” retweets.

After all that processing, the original dataset contains, 1,196 tweets, and for each one, (1) the tweet ID, (2) label, being either true, false, non-rumor or unverified, (3) the tweets’ text, (4) news’ story URL, (5) news’ story title, (6) news’ story content and (7) news’ story content size.

GitHub - dongkwan-kim/HBTP

10 Term frequency – Inverse document frequency (TF-IDF).
Figure 2.2 presents the dataset’s distribution. It plots the used Twitter stories content length as a function of their fact-checked label.

The cleaned dataset is divided into 371, 289, 274 and 262 non-rumor, true, unverified and false stories, see Figure 2.3. For the purposes of this thesis it actually made more sense to reduce the existing labels into just two categories. Unverified tweets were discarded and true and non-rumor labels where tied together into a single category.

Find bellow, on Figure 2.4, the plotted distribution of the used Twitter stories once labels have been reassigned as described. Reducing the number of labels definitively eases the analysis on the plot relationship.

Once label cleaned, the resulting dataset is divided into 660 and 262 true and false
2.2. Our proposal’s approach

stories. See Figure 2.3. Even not extreme, a bias on the used dataset is also clear when labels are cleaned. Data is stripped into 28.4% and 71.6% of false and true stores which has to be taken into account when analyzed and classified. Furthermore, a trend on the relation between label and content size can be drawn based on the previous plots. Apparently true stories tend to be larger than false ones.

Based on this data the architecture described on Section 2.2.2 is implemented to classify them and try to separate false news based on an automatic information process.

Further analysis on the data can be performed from visualizing for each story their content based on the relative importance of each appearing lemma, see Figure 2.5, to a feature extraction or an LDA\(^\text{11}\) modeling visualization. Besides, further detail on LDA based topic modeling uncovering the hidden topics on the dataset is provided on Appendix A.

As per topic modeling, in the context of NLP, is described as a method of uncovering hidden structure on a corpus. It is useful for text classification, recommendation systems and uncovering themes. Figure 2.5, provides a cloud representation of a topic model.

2.2.2 What? How?

This section describes this thesis proposed texts’ classification architecture. It is distinctively splitted into three identifiable pieces: NLP analysis pre-processing, NLP traditional techniques combined with machine learning classification algorithms and a deep learning approach.

Our proposal develops the NLP analysis pre-processing of the Twitter stories data based on the SpaCy NLP toolkit. SpaCy is an open-source software library for advanced NLP published under the MIT licence. It handles many tasks commonly associated with building and end-to-end NLP pipeline, it is written in optimized Cython\(^\text{12}\) being the fastest syntactic parser available.

\(^\text{11}\) Latent Dirichlet Allocation (LDA)

\(^\text{12}\) Programming language written in Python and C, designed to give C-like performance. It is a compiled language that is typically used to generate CPython extension modules.
On the text processing stage, it is lemmatized based on the SpaCy implementation and its’ available large English dictionary. Stop words, punctuation and spaces removal processes are also applied. All along with lemmatization, they are both common to all classification processes implemented.

On the other hand of NLP implemented processes, construction of bigrams and trigrams is only applicable to a subset of the classification techniques. Bigrams are two words, three respectively for trigrams, occurring together in a story piece. *Gensim’s* phrases model can build and implement bigrams, trigrams, quadgrams and more. To apply phrase modeling the following implementation was considered, to determine whether two tokens $A$ and $B$ constitute a phrase.

\[
\frac{\text{count}(A \setminus B) - \text{count}_{\text{min}}}{\text{count}(A) \times \text{count}(B)} \times N > \text{threshold}
\]

where,

- $\text{count}(A)$ and $\text{count}(B)$ are the number of times tokens $A$ or $B$ appear in the corpus.
- $\text{count}(A \setminus B)$ is the number of times the tokens $A \setminus B$ appear in order.
- $N$ is the total size of the corpus vocabulary.

- $\text{count}_{\text{min}}$ and $\text{threshold}$ are user-defined parameters ensuring, the first, a minimum number of occurrences on accepted phrases, and the second, control on how strong the relationship between two tokens is required to be.

Bigrams (or digrams) help provide the conditional probability of a token given the preceding one, when the relation of conditional probability is applied. Bigrams and tri-grams were in fact implemented and 4-grams, even considered and their implementation verified, did not provided additional meaning to our corpus.

Full pre-processed stories; including lemmatization, removal of stop-words, punctuation and spaces, and construction of bi-grams and tri-grams; is feed into a IT-IDF implementation, converting the text for each document into vector models on the basis of occurrence of the words in them. This vectors are them used as input for four machine learning classification algorithms: *Random Forest, k-Nearest Neighbours, SVM* and *XGBoost*. An analogous process is carried out, but excluding the implementation of bi-grams and tri-grams, to apply *Word2Vect* and *Word2Vect* embedding algorithms as input to the classification processes instead of IT-IDF. See Chapter 3 for further detail.

Additionally, a deep learning based implementation was also put up to test including both LSTM and GRU neural networks. Further detail is provided on Chapter 4.

---

13 *en_core_web_lg* pre-trained English statistical models. English multi-task CNN trained on OntoNotes, with GloVe vectors trained on Common Crawl.
Chapter 3

Proposed techniques: Conventional NLP approach

3.1 As a consequence of the "no free lunch" theorem

In mathematical folklore the term "no free lunch" theorem arises from referencing the ideas presented by David Wolpert and William Macready in [76] [75] where they proposed that "any two optimization algorithms are equivalent when their performance is averaged across all possible problems". Besides, Shalev-Shwartz et al. work on [62] is also referred in some cases as "no free lunch" theorem.

**THEOREM** (No-Free lunch, Shalev-Shwartz et al. proposal): Let $A$ be any learning algorithm for the task of binary classification with respect to the $0-1$ loss over a domain $\chi$. Let $m$ be any number smaller than $|\chi|/2$, representing a training set size. Then, there exists a distribution $D$ over $\chi \times \{0, 1\}$ such that:

1. There exists a function $f : \chi \rightarrow \{0, 1\}$ with $L_D(f) = 0$

2. With probability of at least $\frac{1}{2}$ over the choice of $S \sim D^m$ we have that $L_D(A(S)) \geq \frac{1}{8}$.

Which states, as defined on [62] that for every learner, there exists a task on which it fails, even though that task can be successfully learned by another learner.

Following that interpretation, this thesis proposes not one but four machine learning classification algorithms (Random Forest, k-Nearest Neighbours, Support Vector Machine and XGBoost) to address every proposed implementation. Even though, we do not know how to construct the optimal classifier for every given problem, prior knowledge of it may enable a suitable approximation, and therefore on every proposed case, well known algorithms have been tested for all cases.

3.2 Methodology

All hereinafter proposed implementations are preceded by the NLP corpus pre-processing detailed on Section 2.2.2 and taking into consideration the order, limitations and specifications as described on that section.

As an overview, to ease the comprehension of this proposal’s structure, three algorithms are described in this section to address the fake news classification problem in hand, TF-IDF, Word2Vect and Doc2Vec. They will be taking as input the results of
executing the pre-processing techniques described on Section 2.2.2 which in general terms consist on a lemmatization process; the removal of stop words, punctuation and spaces; and the construction of bi-grams and tri-grams where relevant. The first two pre-processing steps are applicable to all implementations described hereinafter on Section 3.2.2 (Word2Vec and Doc2Vec) whereas the last is only applied when executing TF-IDF (described on Section 3.2.1). All three NLP traditional approaches proposed (TF-IDF, Word2Vec and Doc2Vec) are followed by traditional Machine Learning classification algorithms, Random Forest, k-Nearest Neighbours, Support Vector Machine and XGBoost to classify the Twitter stories within the corpus into true and false.

Traditional NLP approaches are described, as previously detailed, on Section 3.2.1 and Section 3.2.2. Since ML classification algorithms will be common to all of them will be presented beforehand. See Figure 3.1 for an overview on their use cases.

**Figure 3.1:** Source: scikit-learn - ML algorithm cheat sheet.

Random Forest is one of the most, if not the most itself, well known decision trees algorithms for ML classification problems. A decision tree, is a hierarchical learning model dividing the observed space based on decision rules. They are a step by step construction method based on a division, stop and pruning criteria. Random Forest, itself, consists on a large number of individual decision trees operating as an ensemble. It is based on trees such that each branch is constructed considering a random attribute’s subset, the best split among that subset is chosen and not pruned trees are used.

Random Forest, was developed and registered by Leo Breiman and Adele Cutler on [7]. However, the first implementation of random decision forest was proposed by Tin Kam How on [32]. However, in contrast with Breinman’s publication [7], the used scikit-learn implementation combines classifiers by averaging their probabilistic prediction, instead of letting each classifier vote for a single class. Random Forest
outperforms boosting algorithms on average, generally avoids over-fitting problems and it is, in fact, one of the more robust classifiers.

On every execution the following parameters’ grid will be inspected though a GridSearchCV. Detail is provided on Table 3.1.

<table>
<thead>
<tr>
<th>Tuned parameters</th>
<th>Tested values</th>
</tr>
</thead>
<tbody>
<tr>
<td>max_depth</td>
<td>10, None</td>
</tr>
<tr>
<td>max_features</td>
<td>1, 4, 10, auto</td>
</tr>
<tr>
<td>min_samples_split</td>
<td>2, 4, 10</td>
</tr>
<tr>
<td>bootstrap</td>
<td>True, False</td>
</tr>
<tr>
<td>criterion</td>
<td>gini, entropy</td>
</tr>
</tbody>
</table>

**Table 3.1: Random Forest - Grid searched parameters**

Regarding k-Nearest Neighbours, hereafter k-NN, it is a type of supervised instance-based learning or non-generalizing learning\(^1\) machine learning algorithm. It does assume that similar things exist in close proximity.

*scikit-learn* proposes two different nearest neighbors classifiers: used *KNeighborsClassifier* implements learning based on the *k* nearest neighbors of each query point, where *k* is one of the grid searched values in the proposed implementation on this thesis. The best fitting value for *k* is actually highly data-dependent, while larger values diminish noise effects, also blur the classification boundaries. Along with the number of neighbours different values on the *weight* and *metric* parameters were put up to test as well. The first defining whether the execution considers uniform or weighted values depending on the distant to the neighbour points, and the second defining the used distance.

On every execution the following parameters’ grid will be inspected though a GridSearchCV. Detail is provided on Table 3.2.

<table>
<thead>
<tr>
<th>Tuned parameters</th>
<th>Tested values</th>
</tr>
</thead>
<tbody>
<tr>
<td>n_neighbors</td>
<td>4, 5, 10</td>
</tr>
<tr>
<td>weights</td>
<td>uniform, distance</td>
</tr>
<tr>
<td>metric</td>
<td>euclidean, manhattan, minkowski</td>
</tr>
</tbody>
</table>

**Table 3.2: k-NN - Grid searched parameters**

Regarding Support Vector Machines, hereinafter SVM\(^2\), they are considered to be one of the most successful machine learning algorithms, relying on three main ideas: linearity, sparseness and kernel trick. The last, as a matter of fact may be the key to their success, since it is one of the features allowing SVM overcoming high dimensional problems, and might be worthy of further detail.

**COVER’S THEOREM** [14]: A complex pattern-classification problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated.

It is one of the primary theoretical motivation for non-linear kernel methods. It is in

---

\(^1\) It does not attempt to construct a general internal model, but stores instances of the training data.

\(^2\) Also called Sparse Kernel Machines.
Chapter 3. Proposed techniques: Conventional NLP approach

fact in the base of how SVM overcome the curse of highly dimensional problems by making the projection only implicitly (thanks to the kernel trick).

In general terms, SVM implementations have been a really successful classification algorithm based on the following properties/advantages: SVM do not have a local minimum, the optimal solution can be found in polynomial time, their implementation is based on a reduce number of free parameters that can be adjusted by cross-validation, they produce stable results and sparse solutions since they only takes into account the support vectors. Besides, maximizing the margin allows to control the complexity independently of the number of dimensions and they have a great generalization capability.

On every execution the following parameters’ grid will be inspected though a GridSearchCV. Detail is provided on Table 3.3.

<table>
<thead>
<tr>
<th>Tuned parameters</th>
<th>Tested values</th>
</tr>
</thead>
<tbody>
<tr>
<td>kernel</td>
<td>rbf</td>
</tr>
<tr>
<td>gamma</td>
<td>$10^{-3}$, $10^{-4}$</td>
</tr>
<tr>
<td>C</td>
<td>1, 10, 100</td>
</tr>
</tbody>
</table>

TABLE 3.3: SVM - Grid searched parameters

Regarding XGBoost, it is a Gradient Boosting based algorithm optimized for speed, allowing to work with highly dispersed data and storing it as to prevent continuously re-arranging attributes. It does include a complexity based lose function penalty:

$$E[L(y, F(x))] + \sum_{m=1}^{M} \Omega(f_m)$$

On every execution the following parameters’ grid will be inspected though a GridSearchCV. Detail is provided on Table 3.4.

<table>
<thead>
<tr>
<th>Tuned parameters</th>
<th>Tested values</th>
</tr>
</thead>
<tbody>
<tr>
<td>learning_rate</td>
<td>0.05, 0.1, 0.2</td>
</tr>
<tr>
<td>n_estimators</td>
<td>10, 75, 150</td>
</tr>
<tr>
<td>max_depth</td>
<td>3, 5, 7</td>
</tr>
<tr>
<td>gamma</td>
<td>0, 0.1, 0.2</td>
</tr>
</tbody>
</table>

TABLE 3.4: XGBoost - Grid searched parameters

XGBoost includes, aside from the lose function penalizing complexity shown before, a great number of parameters to be adjusted. Even if providing great results and an important execution optimized speed it is not easy to parameterize. Initial implementation of Gradient Boosting algorithms was developed by Friedman [20]. Later implementations introduced the idea of an iterative functional gradient descent.

On all the previously mentioned classification algorithms both, roc_auc_score and accuracy metrics are used to calibrate the above mentioned grid searched parameters. Regarding accuracy, in binary classification as it is the problem presented, is currently used as an statistical measure the goodness of a classification, testing how correctly it does identify or exclude a condition. More precisely, accuracy defines the
3.2. Methodology

proportion of true results over the total. It is actually defined as:

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\]

where,

TP = True positive; FP = False positive; TN = True negative; FN = False negative.

Furthermore, the \textit{roc_auc_score} returns the area under the ROC\textsuperscript{3} curve, which, in terms, plots parametrically \( TPR(T) \) versus \( FPR(T) \) with \( T \) as the varying parameter. \( TPR \) (True positive rate) and \( FPR \) (false positive rate) are, respectively, defined as:

True positive rate,

\[
TPR(T) = \int_T^\infty f_1(x) \, dx
\]

and, false positive rate,

\[
FPR(T) = \int_T^\infty f_0(x) \, dx
\]

Assuming, that given a threshold parameter \( T \), a result is classified as "positive" if \( X > T \), and "negative" otherwise. \( \mathcal{X} \) follows a probability density \( f_1(x) \) if the result actually belongs to class "positive", and \( f_0(x) \) if otherwise. Therefore, the area under the curve, provided by the \textit{roc_auc_score} metric can be seen as:

\[
\text{Area} = \int_{x=0}^1 TPR(FPR^{-1}(x)) \, dx
= \int_{-\infty}^{\infty} TPR(T)FPR'(T) \, dT
= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I(T' > T)f_1(T')f_0(T)dT'dT
= P(X_1 > X_0)
\]

Given \( TPR(T) : T \to y(x) \) and \( FPR(T) : T \to x \). Considering that \( X_1 \) is the score for a positive result and \( X_0 \) the score for a negative one.

3.2.1 Token based analysis

Fake news classification problem though NLP token based proposed approach considers the implementation of a Term Frequency – Inverse Document Frequency, here-inafter \textit{TF-IDF}, algorithm whose result will be used as input to the previously described ML classification algorithms. \textit{TF-IDF} is, in fact, and automatic statistical token based approach into classifying word relevance within a corpus.

It breaks down into the two terms that make up its own name,

- \textit{Term Frequency}, can be defined as the number occurrences of a word in a document, normalized by the document’s length.
Chapter 3. Proposed techniques: Conventional NLP approach

- **Inverse Document Frequency**, is inverse to the proportion of documents in the corpus containing each word scaled by the logarithm (aligned with Zipf’s law\(^4\)).

TF-IDF value for a word \(w\), on a specific document \(d\) of a certain corpus \(D\), is given by:

\[
TF-IDF(w, d, D) = TF(w, d) \cdot IDF(d, D)
\]

\[
= \frac{|w \in d|}{|d|} \cdot \log \frac{|D| + 1}{|d \in D : w \in d| + 1}
\]

The actual TF-IDF implementation used for the purposes of this thesis is the standard `scikit-learn`, `TfidfVectorizer` proposed function, where minor changes from the standard textbook notation have to be taken into account.

Detail on the hyper-parameter tuning results based on `GridSearchCV`, based on the previously detailed grids for each of the implemented ML classification algorithms. See Table C.1 on Appendix C.

3.2.2 Embedding based analysis

Two sorts of word embeddings have been applied through this thesis, *Word2Vec* and *Doc2Vec*. Word embeddings are NLP techniques creating real numbers mapping vectors for words in a corpus. In mathematics, an embedding is an instance of some algebraic structure contained within another instance, and as a parallelism from that definition, it “defines” how it takes the embedding from a multi-dimensional space per word to a continuous lower dimension vector space. The underlying concept is that a word is characterized by those “close” to it, and therefore, words with “close” meaning are represented close to each other in the resulting vector-space.

We have loaded and used the *Word2Vec* model created and published in 2013 by Google’s researcher Tomas Mikolov [46], retrieving the trained model’s vector representation for every word in the fed corpus and the averaging all word associated vectors within each story to create a unique vector associated to each Twitter story in the corpus that will be used as an input to the ML classification algorithms described on Section 3.2.

Regarding *Doc2Vec*, this thesis uses the `Gensim` library implementation, providing a unique vector associated to each document in the corpus. Even though, the implementation itself provides a meaningful classification retrieving associated probability to each existing label with remarkable results range accuracy scores over the test set of 85.4%, we have followed the same schema as in previous cases to maintain consistency across experiments.

Detail on the hyper-parameter tuning results based on `GridSearchCV`, based on the parameters’ grids detailed on Section 3.2 for each of the implemented ML classification algorithms. See Table C.2 and Table C.3, for *Word2Vec* and *Doc2Vec* associated parameterization respectively on Appendix C.

---

\(^4\) Empirically introduced by George K. Zipf in the 1940s, states that for a given language the frequency of occurrence of different words follows a distribution that can be approximated by \(P_n \sim 1/n^a\), where \(P_n\) represents the frequency of the \(n\)-th most frequent word, and \(a\) is a real number slightly over 1.
3.3 Testing

For each of the above described algorithms; TF-IDF, Word2Vec and Doc2Vec; the same execution schema has been followed by taken their output results and feeding them into the ML classification algorithms described on Section 3.2. Parameterization of the classification algorithms was performed though a grid search as described in such section, and the results are detailed on Appendix C.

The Twitter based corpus, as described on Section 2.2.1, was split into 80% and 20%, training and testing sets, in order to keep a subset of the data unseen by the classification algorithms that were trained based on the 80% of the data used as training set and their performance evaluated on the resting 20%, based on the following metrics:

- Confusion matrix.
- Classification report, including precision, recall and f1 scores, as well as the support values, for all labels. Additionally, it provides average values including macro average (averaging the unweighted mean per label) and the weighted average (averaging the support-weighted mean per label).
- accuracy score.
- roc_auc score.

Detailed description of both accuracy and roc_auc scores has been already provided on Section 3.2. Regarding the classification report it is the standard report provided on the implementation of scikit-learn, including precision, recall and f1 scores.

Precision and recall, are well related scores, defined as follows:

\[
\text{precision} = \frac{|\{\text{RelevantResults}\} \cap \{\text{RetrievedResults}\}|}{|\{\text{RetrievedResults}\}|}
\]

\[
\text{recall} = \frac{|\{\text{RelevantResults}\} \cap \{\text{RetrievedResults}\}|}{|\{\text{RelevantResults}\}|}
\]

Therefore, precision is understood as the percentage of the retrieved results that are in fact relevant and recall the percentage of the relevant results that were in fact retrieved.

F-1 score relies on the definition of the precision and recall scores. It is, in fact, the harmonic mean of both metrics,

\[
F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
\]

The general formula for any positive real number \(\beta\) is:

\[
F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} = \frac{(1 + \beta^2) \cdot \text{TruePositive}}{(1 + \beta^2) \cdot \text{TruePositive} + \beta^2 \cdot \text{FalseNegative} + \text{FalsePositive}}
\]
The obtained results on each of the implemented algorithm architectures for each of the described metrics will be presented on Section 5.1. However, a table with the comparison of the results in terms of accuracy and roc_auc is presented on Figure 3.5 bellow.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Classification algorithm</th>
<th>Metric</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF-IDF</td>
<td>Random Forest</td>
<td>accuracy</td>
<td>86.49%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>76.85%</td>
</tr>
<tr>
<td>XGBoost</td>
<td></td>
<td>accuracy</td>
<td>87.03%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>81.59%</td>
</tr>
<tr>
<td>k-Nearest Neighbors</td>
<td></td>
<td>accuracy</td>
<td>88.65%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>86.00%</td>
</tr>
<tr>
<td>SVM</td>
<td></td>
<td>accuracy</td>
<td>92.97%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>87.96%</td>
</tr>
<tr>
<td>Word2Vec</td>
<td>Random Forest</td>
<td>accuracy</td>
<td>80.54%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>68.84%</td>
</tr>
<tr>
<td>XGBoost</td>
<td></td>
<td>accuracy</td>
<td>81.08%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>50.00%</td>
</tr>
<tr>
<td>k-Nearest Neighbors</td>
<td></td>
<td>accuracy</td>
<td>81.62%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>71.24%</td>
</tr>
<tr>
<td>SVM</td>
<td></td>
<td>accuracy</td>
<td>70.81%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>50.00%</td>
</tr>
<tr>
<td>Doc2Vec</td>
<td>Random Forest</td>
<td>accuracy</td>
<td>80.00%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>65.74%</td>
</tr>
<tr>
<td>XGBoost</td>
<td></td>
<td>accuracy</td>
<td>85.95%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>77.01%</td>
</tr>
<tr>
<td>k-Nearest Neighbors</td>
<td></td>
<td>accuracy</td>
<td>82.16%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>79.24%</td>
</tr>
<tr>
<td>SVM</td>
<td></td>
<td>accuracy</td>
<td>81.62%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>roc_auc</td>
<td>76.14%</td>
</tr>
</tbody>
</table>

**Table 3.5: Traditional NLP - ML accuracy and roc_auc scores**

The highest obtained results, on both accuracy and roc_auc, where obtained by applying an SVM to the vector space generated after applying TF-IDF to the Twitter stories dataset. Achieving results with 92.97% and 87.96%, accuracy and roc_auc scores respectively, over the test set.
A comparison of the obtained results, by accuracy and roc_auc scores, is presented on Figure 3.2.

**Figure 3.2:** Accuracy and roc_auc scores - Traditional NLP techniques.
Chapter 4

Proposed techniques: Advanced deep learning techniques

4.1 Methodology

First things first, before going into further detail regarding deep learning techniques applied to the fake news detection problem in hand, it might worth the effort trying to clear the confusion and detail the differences telling apart Artificial Intelligence (AI), Machine Learning (ML) and Deep Learning. See on Figure 4.1 a back-of-the-envelope explanation.

![Figure 4.1: Source: technologyreview.com - A back-of-the-envelope AI explainer](image)

Even closely-related, they have a set / sub-set relationship between them being AI the broader and all-encompassing concept set and deep learning the smaller one. A quick definition of each of them might help to clarify.

- Artificial intelligence, as defined by Kaplan et al. on [39] is "a system’s ability to correctly interpret external data, to learn from such data, and to use those learnings to achieve specific goals and tasks through flexible adaptation".

- Machine learning, as defined by Tom Mitchell, "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E".

Deep learning, Deng *et al.* proposed on [17], among others, the following definition: "A class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification.

Altogether, problem solving based on pre-defined rules (*algorithm*), as far as it can be considered "intelligent" behaviour, is what we understand by artificial intelligence. Meanwhile, machine learning, being as subset of it, it is an method of training algorithms so that they can learn the best approach to decision making. Finally, deep learning is just an specific machine learning technique based on the exploit of many non-linear information processing layers.

On the discussed problem of fake news classification, this chapter introduces this thesis’ proposed deep learning approach. Taking, again, the data set described in *Section 2.2.1* and applying the lemmatization and stop words removal processes described in *Section 3.2*, two alternative algorithms’ implementations are proposed and hereafter detailed, on the one hand a LSTM\(^1\) neural network and on the other a GRU\(^2\) neural network.

### 4.1.1 Long Short-Term Memory (LSTM) neural networks

*Long Shot-Term Memory* neural networks, hereinafter LSTM, are, in fact, a sort of artificial *recurrent neural network* (RNN) architecture. RNN networks are based on Rumerhart *et al.* work on [60] in 1986. LSTM on the other hand, were proposed by Hochreiter and Schmidhuber in 1997 on [33].

An LSTM network looks quite like a standard RNN, but for the summation units in the hidden layer being replaced by memory blocks. The following architecture diagram, as provided by Greff *et al.* on [27], a provides clarification on the network’s architecture basis. See Figure 4.2.

---

1. Long Short-Term Memory.
2. Gated Recurrent Units.

---

**Figure 4.2:** Source: Greff *et al.* [27]. - The LSTM unit.
4.1. Methodology

In its original form, LSTM contained only input and output gates. The forget gates, by Gers et al. on [22] and the peephole weights on [23] connecting the gates to the memory cell were further extensions to the original model. The purpose of the forget gates was to provide a way for the memory cells to reset themselves and forget previous input values.

Additionally, having feedback connections they are keen on processing text since it is a sort of sequenced based data. Obviously, its advantages are most favoured when facing problems requiring the use of long range contextual information, such as text classification problems like the one in hand.

Detailed equations regarding both, the forward (activation) and backward (gradient calculation) passes of an LSTM hidden layer are provided on Appendix D.

Among the described advantages of LSTM, preservation of gradient information is one of the more remarkable [40]. Through input gates the network can decide when an input is important enough to be memorized, reset gates allow the network to decide when a memory is no longer useful and output gates when to release a particular memory to compute the current network output. Therefore, a memorized value can be retained indefinitely and thus they do not vanish gradients. They deal quite successfully with the exploding and vanishing gradient problems faced by RNNs.

The implementation presented on this thesis relies on the Keras’ opened libraries. The proposed model is based on a Keras’ sequential model, i.e. a linear stack of layers. A 200 neuron LSTM layer with dropout is presented after a Keras embedding layer. Various implementations were tested modifying the final dense layer activation function.

4.1.2 Gated Recurrent Units (GRU) neural networks

Gated Recurrent Units, hereinafter GRU, neural networks were firstly introduced by Kyunghyun Cho et al. in 2014 on [9]. Further detail on the formulation proposed by Kyunghyun Cho et al. is provided on Appendix D. GRUs have been shown to exhibit even better performance on certain smaller datasets, which as a matter of fact comes as a very useful feature to the problem in hand in this thesis. Performance comparison between LSTM and GRU is, however, a problem still under review. GRU networks, combine write and rest gates into a single update gate with not read gate. They have therefore a lower computational cost.

As a matter of fact, Junyoung Chung et al.[10] provide a valuable evaluation of GRU networks against LSTM neural networks. In fact they concluded that “by using fixed number of parameters for all models on some datasets GRU, can outperform LSTM units both in terms of convergence in CPU time and in terms of parameter updates and generalization. Furthermore, Wenpeng Yin et al. on [81] performed systematic comparison of CNN and the two prevailing RNN, LSTM and GRU, on a wide range of NLP tasks.

GRU neural network presented on this thesis relies as well on the Keras’ opened libraries. The proposed model is based on a Keras’ sequential model, i.e. a linear stack of layers. A 64 neuron GRU layer without dropout is presented after a Keras embedding layer. Various implementations were tested modifying the final dense layer activation function.
A helpful diagram comparing LSTM and GRU units’ architecture, as pictured by Kyunghyun Cho et al.[9] is detailed as well on [10], and provided here for further comprehension. See Figure 4.3.

![Diagram of LSTM and GRU units](image)

**Figure 4.3:** Source: Kyunghyun Cho et al. [9] LSTM vs GRU units.

### 4.2 Testing

First thing to be addressed before testing the proposed LSTM and GRU neural network implementations was leveling the Twitter stories lengths within the dataset. An standardization of the stories length distribution was performed. Based on Figure 4.4 stories were pad using Keras’ `pad_sequence` into same length texts. Sorter texts are padded with value at the end, and longer sequences truncated so that they fit the desired length.

![Twitter stories length distribution](image)

**Figure 4.4:** Twitter stories length distribution

A broad spectrum of padding lengths were tried\(^3\). From lower lengths, closer however to the average length, to higher extreme values. Along with the different padding lengths, a cluster of activation functions, `sigmoid`, `softmax` and `linear` was tested as well.

Achieved results were verified though *StratifiedKFold* performing 10-fold stratified cross-validation. This is a re-sampling technique that can provide a robust estimate of the performance of a machine learning model on unseen data. *scikit-learn* function `cross_val_score` was used to evaluate our model using the cross-validation scheme and print the results. Furthermore, Deep Learning Model Parameters were grid-searched to evaluate different configurations for the proposed neural network model without significant improvements on the achieved scores. See Table 4.1 for further

---

\(^3\) 150, 200, 250, 300, 350, 400, 500 and 600 words padding lengths.
4.2. Testing
detail on the grid-searched parameters and the possible values sequentially tested arrays.

<table>
<thead>
<tr>
<th>Tuned parameters</th>
<th>Tested values</th>
</tr>
</thead>
<tbody>
<tr>
<td>optimizer</td>
<td>SGD, adam</td>
</tr>
<tr>
<td>init</td>
<td>glorot_uniform, normal, uniform</td>
</tr>
<tr>
<td>batches</td>
<td>10, 32, 64</td>
</tr>
<tr>
<td>activation</td>
<td>softmax, relu, linear, sigmoid</td>
</tr>
<tr>
<td>number of LSTM layers</td>
<td>10, 100, 200</td>
</tr>
<tr>
<td>LSTM dropout</td>
<td>0.02, 0.2</td>
</tr>
</tbody>
</table>

Table 4.1: LSTM - Grid searched parameters

Data augmentation processes were tested as well trying to re-balance the dataset in
order to feed both neural networks a more equitably labeled datasets without further
performance improvement.

LSTM neural networks, training/test loss vs accuracy results using a sigmoid activation function are demonstrated on Figure 4.5.

![Loss - Sigmoid](image1)
![Accuracy - Sigmoid](image2)

Figure 4.5: LSTM: Loss/Accuracy scores - sigmoid activation function

LSTM neural networks, using a linear activation function, training/test loss vs accuracy results. See Figure 4.6.

![Loss - Linear](image3)
![Accuracy - Linear](image4)

Figure 4.6: LSTM: Loss/Accuracy scores - linear activation function

Best accuracy scores reported were of 0.7027 with a roc_auc score of 0.5016. Further
tuning, considering all data augmentation, hyper-parameters grid search or K-Folds
validation only increased accuracy up to 0.7081.
Furthermore, GRU neural networks training/test loss vs accuracy results are plotted on Figure 4.7.
Chapter 5

Conclusions and future work

5.1 Conclusions

Based on the implementations described on Chapters 3 and 4, and metrics presented on Chapter 3, the obtained accuracy and roc_auc scores are detailed below for the different algorithm architectures.

\[\text{Figure 5.1: Obtained accuracy and roc_auc scores.}\]

\textit{Figure 5.1} represented scores demonstrate that testing results previously addressed on Chapter 3, even representing old-fashioned techniques, still worth trying, specially when available data is scarce. Best obtained results, based on accuracy and roc_auc scores, for the used dataset were provided by a combination of TF-IDF NLP algorithm and SVM classification implementation, even when more recent techniques, such as text-based embeddings (Word2Vec) or even LSTM and GRU neural networks, were put up to test as well.
Furthermore, detail on the obtained precision, recall and F1 scores is introduced as well in Figure 5.2 below, as part of the performed analysis. The presented information is detailed by each of the tested implementations.

As evidenced by the previous figure, best architecture for a certain problem may highly depend on the task in hand. While accuracy or roc_auc scores define the best overall solution. If an specific result is sought, precision or recall metrics should be contemplated. Those, as defined on Chapter 3, are key when retrieving the maximum data of highest precision on a certain label is the final objective, and based on them, as shown before, the best suited implementation may change.

The obtained results demonstrate that taking into account the complexity level, execution’s times and resources’ requirements, basic algorithms are, at least when minor increases on accuracy are not crucial to the sought outcomes, worth trying and may be in some cases, such as the one in hand, even more accurate. It is however, not a definitively conclusion due to the size and limitations of the used dataset. Fake news classification task it is not the one with the highest available tagged data and therefore unbalanced data is of common usage, this is main reason impeding the deployment of human-out-of-the-loop solutions. Additionally, hyper-parameter tuning for new neural network architectures is definitively not an easy tasks, but highly time consuming and of high computational cost [66].

5.2 Future work

As this thesis was focused on presenting an overview on the main existing text analysis techniques, not deep dive was addressed for none of them. Further hyper-parameter testing could be performed on the presented neural networks as well as their combination with the implemented embeddings.

Additionally, newly introduced transformer network based implementations, such as BERT, are surely worthy exploring when addressing the fake news classification problem. Even though, further data might be necessary to fully exploit the advantages and characteristics of these implementations, the actual fake news and data availability increasing pace will sure lead to an abundance of available data to feed into the different tested strategies.
Appendix A

Topic modelling visualization

Main topic modeling algorithms are Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA) or Latent Semantic Indexing (LSI), Non-Negative Matrix Factorization (NMF) and Biterm topic modeling. This thesis approach was limited to the first as an overall and initial approximation to the modeled corpus.

LDA is a generative statistical model allowing sets of observations to be explained by un-observed groups explaining why some parts of the data are similar. It was developed by Blei et al. in 2003 and it tells us what topics are present in a given document by observing the words in it and producing a topic distribution. By using Bag-of-Words as a method to extract features from our corpus, by means of the Gensim’s LDAMulticore streamed implementation a topic analysis on the dataset can be addressed. Additionally, topic visualization and result interpretation can be performed by means of pyLDAvis interactive LDA visualization python package. It extracts information from a fitted LDA topic model to inform an interactive web-based visualization.

Furthermore, in order to build LDA models and modify the number of topics considered in each case, we have checked the coherence score, based on the gensim implementation.

Analysis based on topic modeling uncovering the hidden topics on the dataset Twitter stories can also be performed. Popular topic modeling algorithms include latent semantic analysis (LSA), hierarchical Dirichlet process (HDP), and latent Dirichlet allocation (LDA), showing the latest excellent results. pyLDAvis is a interactive LDA visualization python package allowing a visualization of the composition of every identified topic, see Figure A.2. Reference from Section 2.2.1.

---

1 Based on Michael Roeder, Andreas Both and Alexander Hinneburg; “Exploring the space of topic coherence measures”
Figure A.2: `pyLDAvis` - Topic visualization.
Appendix B

Implemented algorithm architecture diagram

Detail on the data flow as to the different algorithm architecture is provided below. It is distinctively split into three identifiable pieces: NLP analysis pre-processing, NLP traditional techniques combined with machine learning classification algorithms and a deep learning approach.

**Figure B.1**: Proposed classification algorithm architecture.
Appendix C

Classification algorithms parametrization

Detailed hyper-parameter tuning results based on GridSearchCV for each of the implemented ML classification algorithms.

Parameter results for TF-IDF architecture:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Score metric</th>
<th>Parameterization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>accuracy</td>
<td>bootstrap: False</td>
</tr>
<tr>
<td></td>
<td></td>
<td>criterion: gini</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_features: auto</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min_samples_split: 2</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>bootstrap: False</td>
</tr>
<tr>
<td></td>
<td></td>
<td>criterion: entropy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_features: auto</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min_samples_split: 4</td>
</tr>
<tr>
<td>XGBoost</td>
<td>accuracy</td>
<td>gamma: 0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>learning_rate: 0.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_estimators: 150</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>gamma: 0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>learning_rate: 0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_estimators: 75</td>
</tr>
<tr>
<td>k-Nearest Neighbors</td>
<td>accuracy</td>
<td>metric: euclidean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_neighbors: 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>weights: distance</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>metric: euclidean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_neighbors: 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>weights: distance</td>
</tr>
<tr>
<td>SVM</td>
<td>accuracy</td>
<td>C: 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>kernel: linear</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>C: 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>kernel: linear</td>
</tr>
</tbody>
</table>

Table C.1: TF-IDF - Classification algorithms parameterization
Parameter results for the Word2Vec based architecture:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Score metric</th>
<th>Parameterization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>accuracy</td>
<td>bootstrap: True</td>
</tr>
<tr>
<td></td>
<td></td>
<td>criterion: entropy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_features: 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min_samples_split: 2</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>bootstrap: False</td>
</tr>
<tr>
<td></td>
<td></td>
<td>criterion: entropy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_features: auto</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min_samples_split: 4</td>
</tr>
<tr>
<td>XGBoost</td>
<td>accuracy</td>
<td>gamma: 0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>learning_rate: 0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_estimators: 150</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>gamma: 0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>learning_rate: 0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_estimators: 150</td>
</tr>
<tr>
<td>k-Nearest Neighbors</td>
<td>accuracy</td>
<td>metric: manhattan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_neighbors: 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>weights: distance</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>metric: manhattan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_neighbors: 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>weights: distance</td>
</tr>
<tr>
<td>SVM</td>
<td>accuracy</td>
<td>C: 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gamma: 0.001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>kernel: rbf</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>C: 100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gamma: 0.001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>kernel: rbf</td>
</tr>
</tbody>
</table>

Table C.2: Word2Vec - Classification algorithms parameterization
Parameter results for the *Doc2Vec* based architecture:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Score metric</th>
<th>Parameterization</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Random Forest</strong></td>
<td>accuracy</td>
<td>bootstrap: True</td>
</tr>
<tr>
<td></td>
<td></td>
<td>criterion: entropy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_features: 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min_samples_split: 2</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>bootstrap: False</td>
</tr>
<tr>
<td></td>
<td></td>
<td>criterion: gini</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_features: 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>min_samples_split: 2</td>
</tr>
<tr>
<td><strong>XGBoost</strong></td>
<td>accuracy</td>
<td>gamma: 0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>learning_rate: 0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_estimators: 150</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>gamma: 0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>learning_rate: 0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>max_depth: 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_estimators: 150</td>
</tr>
<tr>
<td><strong>k-Nearest Neighbors</strong></td>
<td>accuracy</td>
<td>metric: manhattan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_neighbors: 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>weights: distance</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>metric: euclidean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>n_neighbors: 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>weights: distance</td>
</tr>
<tr>
<td><strong>SVM</strong></td>
<td>accuracy</td>
<td>C: 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gamma: 0.001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>kernel: rbf</td>
</tr>
<tr>
<td></td>
<td>roc_auc</td>
<td>C: 10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gamma: 0.001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>kernel: rbf</td>
</tr>
</tbody>
</table>

*Table C.3: Doc2Vec - Classification algorithms parameterization*
Appendix D

LSTM and GRU networks underlying equations

D.1 LSTM hidden layer equations

Detailed equations as to how and LSTM neural network models the word sequence $x$, as introduced on [81], is detailed below.

\[ i_t = \sigma(x_tU^i + h_{t-1}W^i + b_i) \]
\[ f_t = \sigma(x_tU^f + h_{t-1}W^f + b_f) \]
\[ o_t = \sigma(x_tU^o + h_{t-1}W^o + b_o) \]
\[ q_t = \tanh(x_tU^q + h_{t-1}W^q + b_q) \]
\[ p_t = f_t \star p_{t-1} + i_t \star q_t \]
\[ h_t = o_t \star \tanh(p_t) \]

where $i_t$, $f_t$ and $o_t$ stand for the three LSTM gates; input, forget and output gates, all generated by a *sigmoid* function over the input ensemble $x_t$ and the preceding hidden state $h_{t-1}$. Additionally, $q_t$ stands for the temporary result generated by a *tanh* non linear operation over the combined $x_t$ and $h_{t-1}$ ensemble, then combined with the history $p_{t-1}$ by input gate $i_t$ and forget gate $f_t$ respectively to get an updated history $p_t$. Finally, output gate $o_t$ over $p_t$ produces the final hidden state $h_t$.

D.2 GRU equations

GRU equations addressing $x$ text sequence modelization, as refereed on [81], is detailed below.

\[ z = \sigma(x_tU^z + h_{t-1}W^z) \]
\[ r = \sigma(x_tU^r + h_{t-1}W^r) \]
\[ s_t = \tanh(x_tU^s + (h_{t-1} \circ r)W^s) \]
\[ h_t = (1 - z) \circ s_t + z \circ h_{t-1} \]

where $x_t$ is and d-dimensional element representing token $x$ at position $t$, i.e. $x_t \in \mathbb{R}^d$, $h_t \in \mathbb{R}^h$ represents the hidden state at position. $z$ and $r$ represent the two gates.
Bibliography


[31] Sardar Hamidian and Mona Diab. “Rumor identification and belief investigation on twitter”. In: Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. 2016, pp. 3–8.


[78] Liang Wu and Huan Liu. “Tracing fake-news footprints: Characterizing social media messages by how they propagate”. In: *Proceedings of the Eleventh ACM


