

Deep Learning Techniques to address FakeNews Detection

Graells Solé, Gabriel
Curs 2020-2021



Director: Freire Veiga, Ana Maria
GRAU EN ENGINYERIA MATEMÀTICA EN CIÈNCIA DE DADES



Universitat
Pompeu Fabra
Barcelona

Escola
d'Enginyeria

Treball de Fi de Grau

Deep Learning Techniques to address Fake News Detection

Gabriel Graells Solé

TREBALL DE FI DE GRAU / ANY 2020

Grau en Enginyeria Matemàtica en Ciència de Dades
Escola d'Enginyeria

Directora
Freire Veiga, Ana Maria



Abstract

Fake news is a growing phenomenon in the digital society characterized by broadcasting hoaxes and false information. The goal of this final project is to study the different methodologies to detect fake news. As part of this work different algorithmic-based solutions are going to be implemented. Particularly, Machine Learning and Deep Learning models will be used to detect hoaxes on the Web.

Resum

Les fake news són un fenomen emergent en la societat digital caracteritzat per la difusió d'enganys i informació falsa. L'objectiu d'aquest projecte és de definir les fake news i estudiar les diferents metodologies per detectar-les. L'estudi implementara diferents solucions algorísmiques. En particular, models matemàtics basats en Machine Learning i Deep Learning seran emprats per a la detecció d'enganys i notícies falses a l'Internet.

Contents

List of figures	5
List of tables	6
1 INTRODUCTION	1
1.1 Project Organization	3
1.2 Work Planning	3
2 BACKGROUND	5
2.1 Deep Learning	5
2.1.1 Activation Functions	8
2.1.2 Dropout Layers	9
2.2 Natural Language Processing	9
2.2.1 Word Embedding	10
2.2.2 LSTM	13
3 LITERATURE REVIEW	16
3.1 State of the art	17
3.1.1 TriFN: Exploits users, authors and news relationships based on mining techniques.	17
3.1.2 Mining Disinformation and Fake News: Concepts, Meth- ods, and Recent Advancement	21
3.1.3 FAKEDETECTOR: Effective Fake News Detection with Deep Diffusive Neural Network	24
3.1.4 dEFEND: A System for Explainable Fake News Detec- tion	30
4 FAKE NEWS DETECTOR	35
4.1 Iteration 1 - Dataset	35
4.1.1 Scraper	35
4.1.2 Exploratory Data Analysis	37

4.1.3	Conclusions	47
4.2	Iteration 2 - Proposed Model	49
4.2.1	Version 1 - LSTM over Titles	51
4.2.2	Version 2 - Authors	54
4.2.3	Version 3 - Tags	61
5	CONCLUSIONS	68

List of Figures

2.1	Perceptron	5
2.2	Feed Forward Network	6
2.3	Convolutional Neural Network	7
2.4	Word2Vec	12
2.5	LSTM	13
3.1	The tri-relationship embedding framework	18
3.2	TriFN optimization problem	19
3.3	Hybrid Feature Learning Unit (HFLU)	25
3.4	Gated Diffusive Unit (GDU)	26
3.5	The Architecture of Framework FAKEDETECTOR	26
3.6	Accuracy Scores	27
3.7	dEFEND Network Diagram	31
4.1	Rating Labels Frequency	38
4.2	Top 10 most Frequent Authors	39
4.3	Pie Plot Top 10 most frequent Authors and un/trustworthiness score	40
4.4	Top 10 Author by TR and UTR scores.	40
4.5	Title Scatter Plot of Terms	43
4.6	Most frequent True/False	45
4.7	Most frequent tags per most frequent authors	46
4.8	Version 1 Diagram	51
4.9	Confusion Matrix Version 1	52
4.10	ROC Curve Version 1	52
4.11	Version 2 Diagram	54
4.12	Confusion Matrix Version 2	55
4.13	ROC Curve Version 2	56
4.14	Confusion Matrix Version 2 True Authors	57
4.15	ROC Curve Version 2 True Authors	58
4.16	Confusion Matrix Version 2 False Authors	59
4.17	ROC Curve Version 2 False Authors	60

4.18	Version 3 Diagram	61
4.19	Confusion Matrix Version 3	63
4.20	ROC Curve Version 3	63
4.21	Confusion Matrix Version 3 True Authors	65
4.22	ROC Curve Version 3 True Authors	65
4.23	Confusion Matrix Version 3 False Authors	66
4.24	ROC Curve Version 3 False Authors	67

List of Tables

4.1	Politifact Scrapped Dataset	37
4.2	Tags Frequency - Number of Tags - Percentage	47
4.3	Version 1 Metrics	51
4.4	Version 2 Metrics	55
4.5	Version 2 Trustworthy Authors Metrics	57
4.6	Version 2 Untrustworthy Authors Metrics	59
4.7	Version 3 Metrics	62
4.8	Version 3 Trustworthy Authors Metrics	64
4.9	Version 3 Untrustworthy Authors Metrics	66

Chapter 1

INTRODUCTION

Fake news is a growing phenomenon in the digital era which creates a perfect technological ecosystem that provides a high throughput channel for broadcasting misinformation. Fake news is not a creation of the digital generation of this century. The repetition of ideas to deceive large volumes of people has been present since the beginning of human history. A notorious case of deliberate spreading of misinformation happens in the Roman Republic during a civil war between Octavian and Mark Anthony. In order to win, the people of the Octavian empire spread the hoax that Mark Anthony had an affair with Cleopatra, an act completely against Roman social morality. He used the contemporary channels of fast transmission, coin engraving, poems, and songs which are the equivalent of modern Facebook or Twitter. Eventually, Octavian became a popular leader winning the war which made him the emperor for over 40 years[4].

A lot can be learned from this first known appearance of fake news. The spreading of fake news usually has the motivation to deceive a large number of the population into believing a hoax the benefits the publisher of the misinformation. The success of the broadcasting and deceiving process does not rely on factual claims rather than on the repetition of the lie. The misattributed phrase to Joseph Goebbels, the propaganda minister in Nazi Germany, also shows this fact: “If you repeat a lie often enough, people will believe it, and you will even come to believe it yourself.” [8].

In our current Information Society, fake news is a major thread[24]. After the creation of the Internet and the easy and fast access to information, truth has become a malleable body that can be shaped as one pleases if

the required procedures are followed. This thread plays a major role in society as people are being manipulated to alter elections results with the use of massive misinformation broadcasting. A clear example of the deceiving capabilities of fake news happened in the USA in 2016 where many citizens based their voting decisions on the constant bombarding of fake news on social media. Regardless of the results of the ballot, those elections became a constant fight between parties that outcome a polarized society where the individual polarization was highly conditioned by hoaxes and misinformation. The disruption of the concept of truth and the disgusting treatment of it denigrated democracy and showed that repetition beats factual information as the general acceptance of an idea becomes fact. Pressure group theory[3] explains this transformation from false information to social truth. In the seek of acceptance the human accepts whatever society claims to be true, the more people that sustain a false belief as a truth the more people engage with it, and the social truth strengthens.

The goal of this work is to study mathematical and programming methodologies that can prevent the creation of fake news. Technology created a perfect environment for the proliferation of fake information and thus, technology must solve this massive thread. To do so, fast and agile methodologies on fake news detection must be implemented to identify early-stage false information. A rapid detection prevented the creation of social fallacies where a piece of false information is sustained by repetition and the general acceptance of it.

1.1 Project Organization

This project structure starts with a brief introduction in the subsequent chapter 2 named Background. In the Background chapter 2, a collection of modeling concepts is presented in order to familiarize the reader with concepts that will appear in coming chapters.

The chapter after Background is named Literature Review chapter 3. The objective of this chapter is to demonstrate the research done previously to the implementation of the codes. Only a small portion of the entire research is shown in this section. The criteria behind the election are the relevance of the presented work and as well as the approach used in those papers in order to maximize the number of techniques reviewed. Two of the reviewed paper's codes have been reproduced and are accessible via a link to a Github repository.

Chapter four focuses on the practical work done in this final project. Here the implementations are discussed and results are studied in depth. It constitutes the core chapter of the project as it displays most of the programming, analytical, mathematical, and problem-solving skills.

1.2 Work Planning

The project work plan has been organized into three main workloads. The first workload consisted of information gathering. As its name indicates its purpose was to collect as many sources of information as possible to create the knowledge foundation of the TFG.

The next workload inherited from the previous workload as its purpose was to read and study all the sources collected. After the studying, only relevant information had been preserved, a small portion of it can be seen in this memoir. The selection of this information also included which article's code will be reproduced and defined in general terms how my own model architecture would take form.

The final workload consisted of two main tasks. The implementation of the different models as well as the writing of the memoir. The final workload timeline has been as follows. First, the failed attempt to implement the code in the article "*FAKEDETECTOR: Effective Fake News Detection with Deep Diffusive Neural Network*" [38] which serve as an introduction to the Python

library TensorFlow ¹. Then I decided to implement my proposed model using the dataset scrapped for the reproduction of the previously mentioned paper. The implementation of the proposed model introduced to the Python library PyTorch ² and help me to develop my Deep Learning modeling skills.

After these two implementations, I started writing this memoir. When the memoir was almost done I begin coding the reproduction of the code proposed in the paper “*dEFEND: A System for Explainable Fake News Detection*” [9]. The motivation behind reproducing this code was the usage of people’s interaction with news in social media and as well as the explainability aspect of this model. Explainability is an emerging topic in ML/DL work as it breaks the black box [6] quality inherent to this type of modeling. On top of that, the classification task of a fake news detector must require justification, an aspect missing in most of the state-of-the-art related work.

¹TensorFlow: <https://www.gossipcop.com/>

²PyTorch: <https://pytorch.org>

Chapter 2

BACKGROUND

In this section, different Machine Learning and Deep Learning algorithms and concepts will be explained. This is a preparation for the following sections where these concepts will be used in the different model's implementation.

2.1 Deep Learning

Deep learning is a family of methods based on artificial neural networks. Its learning models are based on large networks of interconnected perceptrons which receive numerical inputs which codify the data. Perceptrons process the data and interchange information between them to produce an output in a form of a prediction.

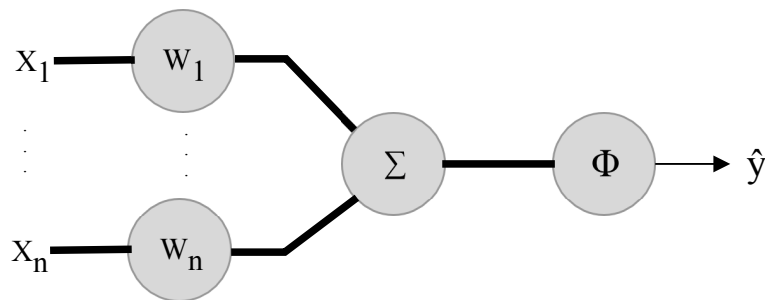


Figure 2.1: Perceptron

The predictions can be grouped into regression, classification, and clustering. Regression estimates the target value y . This technique is used for different applications such as stock price prediction, pricing, risk management, within others. Clustering aggregates data points using as a criterion a particular metric. Classification uses input features to predict the label associated. The model for detecting fake news will be classifier. For this project the input data will be mainly formed by textual and categorical data types. The label which the model is going to predict is either true or false, this labels indicate if a news article contains false information or true.

Deep learning neural networks can also be classified by the connectivity between perceptrons or neurons. Three main categories are defined, feed-forward, convolutional and recurrent. Feed-forward neural networks are characterized by the fact that connections between neurons do not cycle, which implies that information flows in one direction from the input layer to the output layer generating a directly acyclic graph. This is the base model for neural networks.

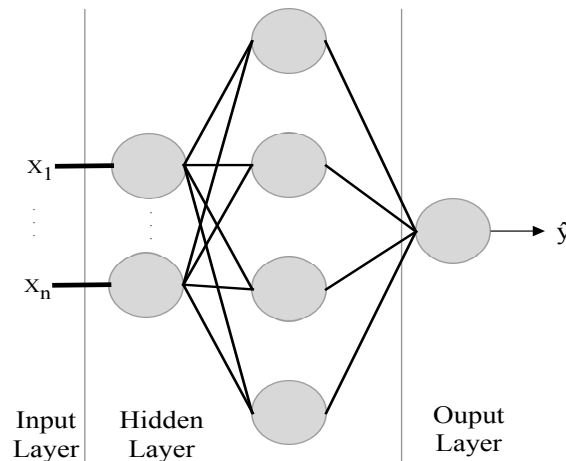


Figure 2.2: Feed Forward Network

Categorical input data is mainly processed using a perceptron learning architecture similar to the figure2.2.

Convolutional networks (CNN) unlike basic feed-forward networks perceptrons do not process the entire input data. Instead each perceptron focuses on a particular region of the input data.

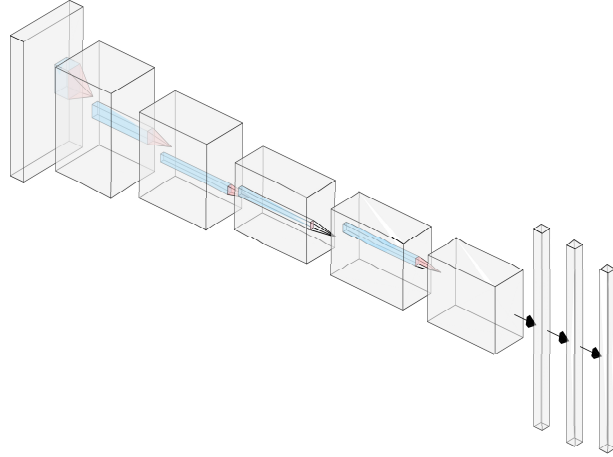


Figure 2.3: Convolutional Neural Network

These methodologies are mainly used in image and video processing tasks as they resemble the organization of the visual cortex in animals [22]. There applications of CNN in NLP (Natural Language Processing) tasks but usually are applied to large textual inputs. Even though these methodologies could be used it has been decided to not form part of this project since it is not best suited for this application.

Finally, recurrent neural networks (RNN) are characterized for having previous outputs as inputs of neurons. LSTM (Long Short-Term Memory) is a particular cell or neuron used in RNN. RNN is used for time series data and as justified in the upcoming subsection NLP this structure is best suited for textual input features.

Then it is concluded that, in general, the models to be implemented will be feed-forward using basic perceptrons for the categorical inputs. RNN, particularly LSTM [19] and GRU (Gated Recurrent Unit) [7], structures will be used for processing time-series data, textual information.

2.1.1 Activation Functions

Activation functions are a collection of mappings that transform the output of the weighted sum found in all cells to a defined range of values. In this section, only the activation functions found in the project code implementations are going to be explained. Those being Sigmoid and ReLu.

Sigmoid

This layer applies a simple weighted sum to the input x .

$$y = xA^T + b$$

Then the activation function sigmoid is applied to y producing the final output. Sigmoid is a function that clamps input values between the range $[0, 1]$.

$$\sigma(y) = \frac{1}{1 + e^{-y}}$$

If the dimensionality of the output of the linear layer is 1 then when the activation function is applied, a value between $[0, 1]$ is output by the model. This corresponds to the confidence level of the model on whether the given news is true or false. Outputs such that $\hat{y} < 0.5$ are considered positive while $\hat{y} \geq 0.5$ are considered negative. [28]

ReLU

Categorical variables can be easily treated with a linear layer consisting of a weighted sum.

$$y = xA^T + b$$

The output vector is then passed to the activation function ReLU. ReLU or Rectified Linear Unit has a range of values $[0, \infty)$ and it has the following mathematical form.

$$R(y) = \max(0, y)$$

There is no mathematical prove why to use ReLU or any other activation function in particular. Usually trial and error or intuition with vague mathematical motivation is the decision behind any activation function implementation. In this project trial and error was been the method to justify the usage of a ReLU activation function. [5]

2.1.2 Dropout Layers

The dropout layer is used mainly in small training dataset models to prevent overfitting. Overfitting is a phenomenon in ML/DL models which happens when the model adapts to the training dataset but fails to generalize. This effect is detected when the accuracy for the training dataset is greater than the validation or test datasets accuracies. To prevent this undesired outcome some regularization techniques such as dropout are applied. Dropout randomly transforms some of the entries of the input vector to 0 with a probability p following Bernoulli distribution[17].

2.2 Natural Language Processing

Natural Language Processing or NLP is a collection of Machine Learning and Deep Learning models and algorithms which aim to write programs with the ability to understand the human language. The current state-of-the-art work is capable of generating, translating languages, speech recognition and, embedding within others.

Given the textual nature of the news used in this work, NLP is one of the core subfields which influence this project implementations.

NLP had its initial form as Symbolic NLP which started with the work by W. John Hutchins in the paper *“The Georgetown-IBM Experiment Demonstrated”*[20] a translator between Russian and English. Symbolic NLP was based on hard-coded rules. With the increase of computational power, NLP systems evolved to Statistical NLP. One of the early works in this area is *“Foundations of statistical natural language processing”*[23] by Manning C, Schütze H. Some of the techniques used are Bayesian networks and Markov networks, these methodologies are considered Machine Learning. These models are currently used in simple and compact text completion on many mobile applications.

The current approach to solve NLP problems is based on Deep Learning techniques such as RNN, LSTM and Transformers which have strong similarities in structure with their predecessors Bayesian and Markov networks. Unlike their predecessors, RNN and LSTM managed to capture the complexity of language at a superior level since these models can detect patterns related to abstract linguistic concepts such as sarcasm.

Recurrent Neural Networks was first introduced by David E. Rumelhart,

Geoffrey E. Hinton Ronald J. Williams in the research work "*Learning representations by back-propagating errors*"[30].

A huge set of variations emerged from this initial work, one groundbreaking variation was LSTM. Long short-term memory (LSTM) architectures solved an intrinsic limitation of RNN, the vanishing gradient problem, which was solved in the scientific publication "*Long Short-term Memory*" by Sepp Hochreiter and Jürgen Schmidhuber [18]. Heuristically, the vanishing gradient problem prevented RNN architectures from maintaining long-term dependencies on time series data which implies that intricate information codified in a one-page long text, per se, would not be captured by the model.

Transformers [35] is another kind of architecture that improves long dependencies and allows parallelization. Transformers also address the already commented problem of long-term dependencies in textual information by introducing an Attention mechanism. In short, transformers break the sequential processing of time-series data which has intrinsic problems which lead to the vanishing gradient problem. As the original scientific article title suggests "*Attention is all you need*" [35] which heuristically implies that the element 0 in the sequence does not have to have relevance nor dependency with the first subsequent element. By shifting the model from sequential to parallel it can focus on the relevance or dependency of an element i to the element j which may not be right after element i . This is an extremely brief and heuristic explanation as in this project the only used element in the paper [35] is the attention layer in the reproduction of the model dEFEND [9]. Further explanation on the Attention layer can be found in the discussion section of the dEFEND [9] model.

The textual information in a news article is a time-series data and thus must be processed with one of the previously mentioned techniques. Due to its superior performance LSTM architecture as well as Attention layer in Transformers will be used for the processing of this data type.

2.2.1 Word Embedding

Word embedding is a widely used technique in NLP. The core idea behind word embedding is to transform terms codified as either one-hot or numerical encoding. A simple yet brilliant architecture is used to achieve this transformation. [25] [29]

There are many variations of word embedding but this section focuses on

the one used in the implementation. CBOW is the architecture used in the `PyTorch` function `Embedding`¹

The input of the model is the one-hot presentation of a term, a vector x of size V , where V is the size of the dictionary containing all the words. Then the input will have $V - 1$ units with value 0 and one unit representative of the word with value 1. The hidden layer weights is a $V \times N$ matrix, each row is a N dimension vector v_w associated with a word w . This association is because when multiplying the input vector by the hidden weight matrix all of the rows will be multiplied by 0, thus becoming 0 except for the row w which corresponds to the dimension k where the 1 in the input is found.

$$h = W^T x = W_{k,.}^T := v_{w_I}^T$$

No activation function is used in the hidden layer. The output layer is a weighted sum with matrix W' and dimensions $N \times V$. The activation function is a softmax function, to obtain the posterior distribution of words, which is a multinomial distribution.

$$p(w_j | w_I) = y_j = \frac{\exp(u_j)}{\sum (\exp(u'_j))}$$

Where u_j is the output of the weighted sum, $u_j = v'_{w_j}^T h$. Thus the output layer is the posterior probability of each term given the input term. The loss function is simply the subtraction between the output probabilities by 0 for all terms and by 1 for the input term.

¹PyTorch Word Embedding: <https://pytorch.org/docs/stable/generated/torch.nn.Embedding.html>

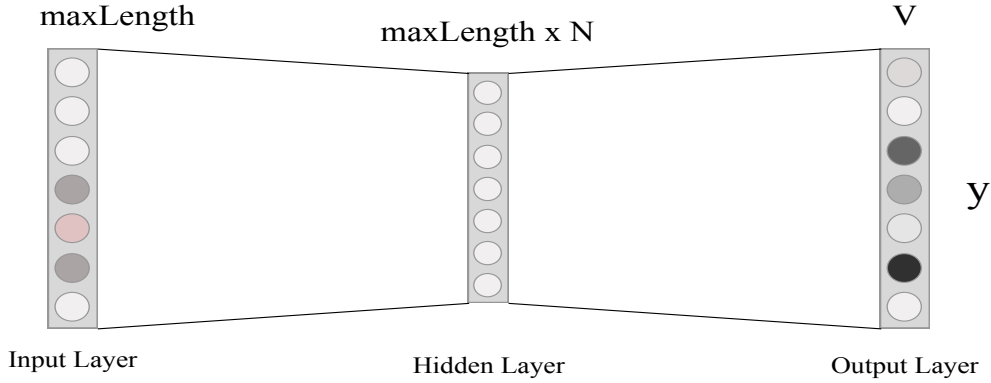


Figure 2.4: Word2Vec

The key aspect of this model is that we are not concerned with the output layer, instead, we find the vector representation of a word w in the hidden layer weights row vector v_w . Then matrix W contains the vector representations for all V words.

To capture better the contextual meaning of a word this method can be adapted by using a window of words. Given the sentence:

*“recurrent neural network **shares** the same weights across several time steps”*

The term “*shares*” is the target word and the terms “*neural*”, “*network*”, “*the*” and, “*same*” are the contextual terms. All of the terms are one-hot encoded and the vectors are averaged to construct the input term.

$$x = \frac{1}{C}(x_1 + x_2 + \dots + x_C)$$

The rest of the network’s functionality is maintained then the target in the loss function is x .

Preprocessing of text

Some preliminary steps are required to prepare the textual information to be used by the models as well as to curate and reduce the disparity of terms.

First, to reduce the number of terms, a set of preprocessing steps are applied to the term in each textual input to output a curated transformation of the original input. All of the terms are transformed to lowercase and all punctuation symbols are removed. To reduce the dimensionality all

meaningless terms such as articles or connectors are removed, the dictionary of stopwords to be removed comes from the NLTK² Python library. Finally, a stemming step is applied. Stemming maps all words to their root term, e.g. *consult*, *consultant*, *consulting*, *consultatory* \rightarrow *consult*. NLTK stemmer function has been used for stemming.

After this preliminary step, a dictionary of words appearing in the textual data inputs is created where, for each word a number is given, e.g. "hello":0, "world": 1,... Lastly, all string are transformed to their numerical representation by substituting each term by the associated number. A maximum length must be defined, using the following metric $max_length = \mu + 2\sigma$ where μ and σ are the mean and the standard deviation of the length distribution of titles. The maximum length is defined, textual inputs with fewer terms are padded with 0 at the end, titles with more than the maximum length are chopped the fit maximum length. Now the textual feature is ready to be processed by the model.

2.2.2 LSTM

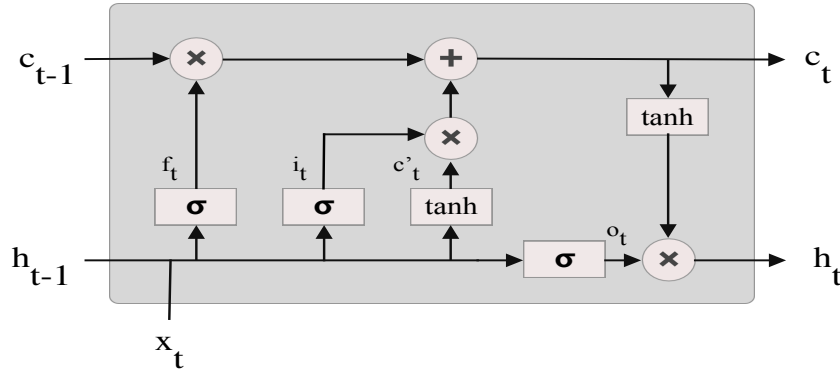


Figure 2.5: LSTM

LSTM architecture can be explained heuristically by labeling each of the paths in the diagram. But first, let's see what are the data inputs of the cell. c_{t-1} is the previous cell state, h_{t-1} is the previous cell hidden state, and x_t is the current input data, an embedded work.

²NLTK: <https://www.nltk.org/>

- **Forget Gate**

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$

As its name indicate it is responsible for deciding what information should be kept or forgotten in both the previous hidden state and the current input data. This allows the network to use only useful information, for example, connectors will be forgotten as those terms do not add information. Mathematically, forgetting happens when the activation function sigmoid maps values to 0. On the opposite, values mapped to 1 are extremely relevant terms. There is a gradient of importance in the range of values from 0 to 1.

- **Input Gate**

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$

The first path of the input gate is equal to the forget gate in structure. The second path of the input gate uses as activation function \tanh which squishes values between -1 to 1 , the output is known as candidate. Finally, the input gates and the candidate gate are element-wise multiplied.

- **Cell State**

$$\tilde{c}_t = \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t$$

Cell gate produces the current cell state output.

- **Output gate**

$$h_t = o_t \circ \sigma_h(c_t)$$

It produces the current hidden state. This cell is used to produce the output of the current cell and it is also passed as input to the next cell.

The key aspect of this architecture is *parameter sharing* between cells. When applied to textual data parameter sharing implies that the relevance of a term not only depends on the term itself but also on the previously seen terms by them model, like this, LSTM cells manage to capture the intricate information codified in a set of terms.

All of the hidden state outputs are concatenated to form a single vector which is passed to the subsequent layers of the model.

Chapter 3

LITERATURE REVIEW

In this section different scientific articles are reviewed and discussed. This serves as a preparation for the different implementations since it introduced me to the different modeling approaches to detect fake news.

The articles reviewed in this chapter represent a small subset of the wide study done. The selection has been based on these norms. The relevance of the article given the objectives of this project. The potential implementation of the algorithm based on the easy access to the used datasets and the computational complexity of the algorithms. Since two of them had been reproduced the computational complexity must be suited for commodity hardware. Some of the reviewed articles not appearing in the project such as *Grover* [36] required enormous computational power as it is based on *GPT-2* [?] model architecture.

One of the articles appearing in this section does not follow the previously mentioned norms, the article Mining Disinformation and *Fake News: Concepts, Methods, and Recent Advancement* [32]. It has been selected since it summarizes and presents in a clear and justified manner the current state-of-the-art work for fake news detection.

3.1 State of the art

3.1.1 TriFN: Exploits users, authors and news relationships based on mining techniques.

Kai Shu, Suhang Wang and Huan Liu [33]

This scientific paper identifies three elements appearing in news posted on social media, publishers, news and, social engagements. These three elements interact with each other and the study of them together provides further information compared to an individualistic study to assess the credibility of the news. The implementation of this paper is an Embedding ML program that uses an optimization using *alternating least squares* to extract the *latent factors* of the proposed problem. A set of matrices are proposed to capture the interaction between the three previously mentioned elements. Defining first the three elements as:

- Publisher P ; $|P| = l$
- News article A ; $|A| = n$
- Users, U ; $|U| = m$

The interaction between elements is represented by:

- X : News article feature matrix $X \in n \times t$

- A : user adjacency matrix $A^{m \times m} \in \{0, 1\}$

Where 1 in index position (i, j) indicates that user i and user j are friends.

- W : social media engagement matrix $W^{m \times n} \in \{0, 1\}$

Where 1 in index position (i, j) indicates that user i has interacted with news j .

- B : publisher news relation matrix $B^{l \times n} \in \{0, 1\}$

Where 1 in index position (i, j) indicates that publisher i has published news j .

- o : partisan vector publisher $o \in \{-1, 0, 1\}$ -1 : *left*, 0 : *neutral*, 1 : *right*

The diagram below shows the latent factors extracted from the previous matrices and how they interact with each other.

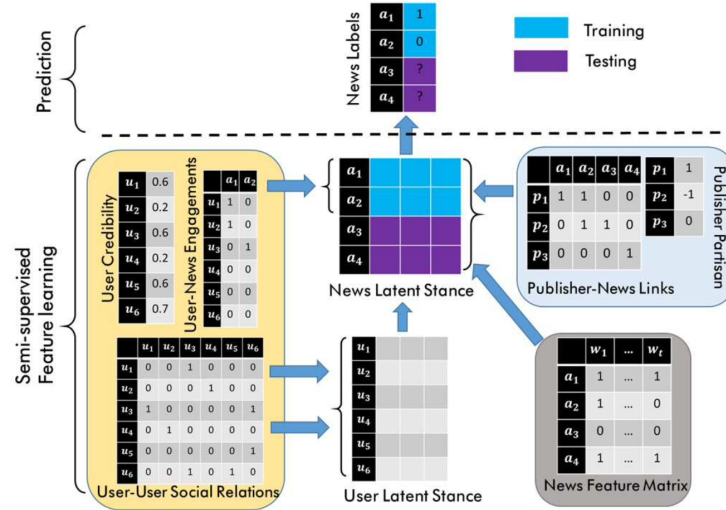


Figure 3.1: The tri-relationship embedding framework [33]

As mentioned above the problem is solved by extracting the latent features codified in the matrices. To do so nonnegative matrix factorization is used to extract the latent features in news content and for news-user social engagement matrix. For user engagement and publisher partisan matrices, an optimization Laplacian problem is proposed to extract the latent features. The combination of all optimizations generates the following optimization problem:

$$\begin{aligned}
& \min_{\theta} \underbrace{\|\mathbf{X} - \mathbf{D}\mathbf{V}^T\|_F^2}_{\text{News Feature Learning}} + \underbrace{\alpha\|\mathbf{Y} \odot (\mathbf{A} - \mathbf{U}\mathbf{T}\mathbf{U}^T)\|_F^2}_{\text{User-User Relation Modeling}} \\
& + \underbrace{\beta\text{tr}(\mathbf{H}^T\mathbf{L}\mathbf{H})}_{\text{User-News Engagement Modeling}} \\
& + \underbrace{\gamma\|\bar{\mathbf{B}}\mathbf{D}\mathbf{Q} - \mathbf{o}\|_2^2}_{\text{News Publisher Partisan Modeling}} + \underbrace{\eta\|\mathbf{D}_L\mathbf{P} - \mathbf{y}_L\|_2^2}_{\text{Fake News Prediction}} \\
& + \lambda(\|\mathbf{D}\|_F^2 + \|\mathbf{V}\|_F^2 + \|\mathbf{U}\|_F^2 + \|\mathbf{T}\|_F^2 + \|\mathbf{P}\|_2^2 + \|\mathbf{Q}\|_2^2) \\
& s.t. \quad \mathbf{D}, \mathbf{U}, \mathbf{V}, \mathbf{T} \geq 0
\end{aligned}$$

Figure 3.2: TriFN optimization problem [33]

Discussion

This paper pioneered using external sources of information not strictly contained in a news article to detect fake news in social media. The proposed problems show great accuracy of almost 0.9 in a binary classification task. Some intrinsic issues are present in this approach. First, the latent feature optimization problems fail to generalize in general for two reasons, it is strictly adapted to the input dataset failing when particular news has users and publishers not appearing in the initial training dataset. On top of that, constructing a model that generalizes is non-viable computationally as the matrices will have huge dimensionality. Using user interactions, partisan scores are brilliant heuristics to guide the algorithm in the classification but it requires huge extra computational resources. Social media is characterized by an immense amount of data traffic. Detecting which users are worth

studying, the relations between them and the publisher, and defining a partisan score in a news social media post can be a rather cumbersome task. These two threads make this implementation not generalizable and thus it has no application in the real world. Nevertheless, it is a good first step towards a fake news detection system.

3.1.2 Mining Disinformation and Fake News: Concepts, Methods, and Recent Advancement

Kai Shu, Suhang Wang and Huan Liu [32]

The paper proposed the implementation of a DL/ML model based on weak supervised learning motivated by the lacking of a consistent and well-formed dataset on fake news.

Weak supervised learning is a machine learning branch that proposes methods for poorly labeled or unlabeled data with the goal of producing labels based on heuristics provided by external knowledge bases, patterns/rules, or other classifiers[2].

The heuristics used for the creation of a solid fake news dataset are motivated and justified by social theorems that characterize and patronize human behaviors. The main social theorem which derives the set of social networking study heuristics used in the labeling of data is the *social homophily theorem*. The *social homophily theorem* states that individuals tend to interact and socialize with other like-minded individuals this effect can be appreciated in the spread of news on social media platforms. The existence of the social homophily theorem generates the *echo chamber effect* where the repetition of news content within the same social circle amplifies and reinforces a particular belief. These two social theorems have strong implications for the proposed task of heuristically labeling a dataset by inferring external information not directly contained in news published in a social media channel. Thus it is expected that fake news in social media have at least two strongly identified social groups that interact with it. One of the groups will defend the claims while the other will be completely against it. People with strong political bias tend to reflect *confirmation bias effect* they tend to accept as true affirmations that confirm their beliefs. These three social theorems are used to construct and justify the following set of heuristics.

News with conflicting viewpoints or sentiments is more likely to be fake news. Publishers and users with a high degree of bias are more likely to publish disinformation Users with low credibility are more likely to publish disinformation while users with high credibility are less likely to publish disinformation.

From these three points, the heuristics are derived.

- The first point derives from the study of sentiment analysis over news post comments. If a news piece has a standard deviation of user sentiment scores greater than a threshold τ_1 , then the news is weakly labeled as fake news.
- The second point is used to label publishers and users bias with a score between -1 to 1 . If the average value of user absolute bias scores is greater than threshold τ_2 , then the news pieces are weakly labeled as fake news.
- Finally, the third point derives the following rule. If a news piece has an average credibility score less than a threshold τ_3 , then the news is weakly labeled as fake news.

The paper provides interesting intuitions about social networking and bias studying of fake news in social media but does not provide implicit implementations. This article has been great guidance in the understanding of fake news and the broadcasting and the intrinsic social interactions in this kind of disinformation.

Discussion

Some crucial and evident aspects are discussed in the paper. The first important realization is that the detection of fake news based entirely on the news article content such as the title, body, publisher, and authors are not enough to construct a robust classifier. Two reasons are presented, first, it is observed that not all sentences in news contents are fake, and in fact, many sentences are true but only for supporting wrong claim sentences. This implies that only *stylistic biases* could be used as a criterion for the classifier. Secondly, the only procedure to prove the falsity of news is by *proof by contradiction*, which means to use another source of information that denies the news claims. This is impossible to achieve by only inputting the news itself into the model.

These facts motivate the use of external sources to input the classifier previously presented. But there are some other aspects important for fake news detection that the proposed methods are lacking the early warning of disinformation. The detection of fake news with the proposed heuristics fails

in a crucial task, detect false information in an early stage to prevent the spread of them leading to *confirmation bias effect* together with the *echo chamber effect*.

Another big challenge is discriminating which users interacting with a news post are worth studying, news posts can get up to million interactions which creates a huge computational load if all users have to be analyzed. Maintaining a database of users and scores can be a storage and computational challenge that affects the viability of this kind of implementation.

3.1.3 FAKEDETECTOR: Effective Fake News Detection with Deep Diffusive Neural Network

Jiawei Zhang, Bowen Dong, Philip S. Yu [38]

This scientific paper focuses on the study of the textual information provided by the news. Three main elements are imputed as an information source for the classification algorithm, which is, the title, the body, the author, and the subjects in form of tags that the news is referring to. These four features are imposed by the scrapped dataset from the fact check organization Politifact [1]. Politifact is a non-profit organization where volunteers rate claims from US political agents by providing arguments to defend or oppose a claim.

The proposed classification model lays within Deep Learning categorization, more specifically, in NLP. They use particular innovating network connectivity known as Deep Diffusive Neural Networks first implemented in the paper, *Get Rid of Suspended Animation Problem: Deep Diffusive Neural Network on Graph Semi-Supervised Classification* [37].

The elements and architecture of the model are as follows.

- Explicit feature extraction

Besides the shared terms in title and body, authors and subjects there are some elements strictly correlated with either false or true news. They extract those terms to generate a one-hot encoding where each time input news has a term in the list of selected terms the particular index associated with that term is set to 1.

- Latent feature extraction

On top of directly correlated terms, there are some hidden signals are codified in the news feature. The goal of the layer is to extract and detect those hidden patterns. A hidden GRU layer followed by a linear layer and a sigmoid activation function is used to detect those patterns.

$$h_{i,t} = GRU(h_{i,t-1}, x_{i,t}; W)$$

$$x_{i,t}^l = \sigma\left(\sum_{t=1}^q W_i h_{i,t}\right)$$

Both Explicit features and Latent features are concatenated. The original input had the following shape $x = [(title, body), subject, author]$ after these two layers $x' = [((title, body)_e, (title, body)_l), (subject_e, subject_l), (author_e, author_l)]$, where the subscript e indicated explicit feature and l latent feature.

Both Explicit and Latent features are combined in a single cell named HLFU, Hybrid Learning Feature Unit, as depicted in the diagram below 3.3.

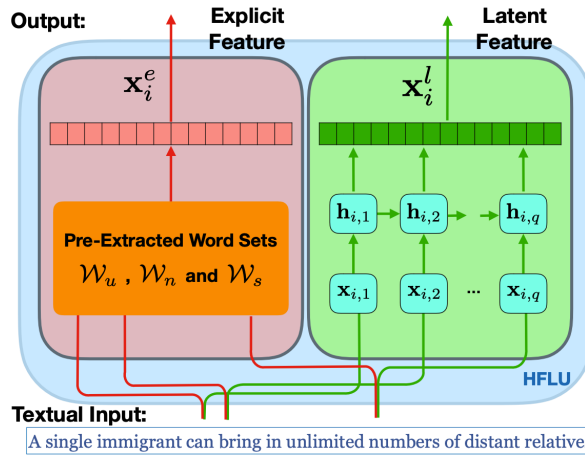


Figure 3.3: Hybrid Feature Learning Unit (HFLU)
[38]

- Deep Diffusive Unit Model

The goal of this architecture is to not only detect the patterns codified within a feature but also to detect patterns in the interaction between features of the same news. A particular author may have a strong bias over a subject which may lead to claiming disinformation. In this case, these two features interact with each other codifying further information than the features on their own. The Deep Diffusive Unit is an RNN cell that has gates seen in well-known RNN cells such as LSTM. Those gates are the forget gate and the adjust gate. The diagram below shows the architecture and connectivity of a Deep Diffusive Unit cell named GDU.

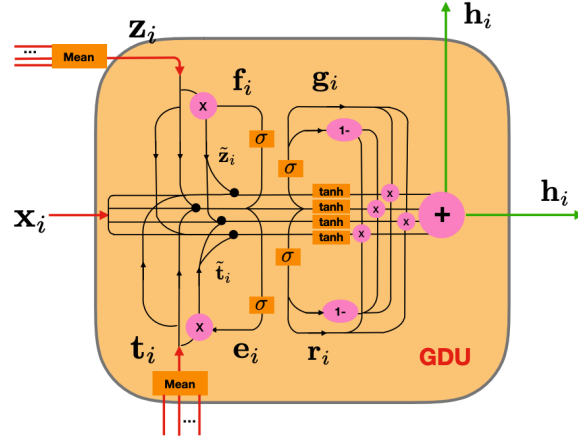


Figure 3.4: Gated Diffusive Unit (GDU)
[38]

HFLU together with GDU forms the basic elements of the network. The overall architecture looks like so:

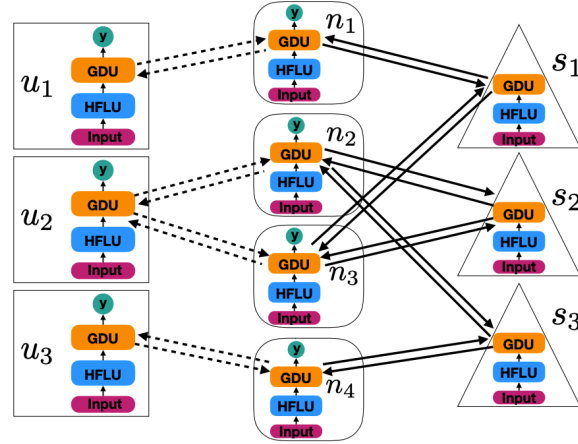


Figure 3.5: The Architecture of Framework FAKEDETECTOR
[38]

Letter u represents the author's feature, letter n represents the title and the body concatenated and, letter s represents the subjects. Note that both the authors, the title/body and, the outcome of the subject independently a prediction on whether the news is false or true. Thus four different losses are computed, one for each feature and the last one which is the accumulated

loss of the three features. (The diagram 3.5 does not show output for the subject but in the mathematical formulation of the paper as well in the metrics section the subject loss exists.)

Discussion

Contrasting other papers FAKEDETECTOR focuses on a classification task based entirely on the news content. As discussed in previous reviews the detection of fake news entirely based on news content is a rather cumbersome task as the model can only detect stylistic bias or contradictions within the same news which may be used in the classification assessment. In fact, the metric results on the paper confirm this claim. Since the input is composed of three features and each feature outputs a prediction the metrics can be applied to each of the feature outputs to evaluate the impact in the classification.

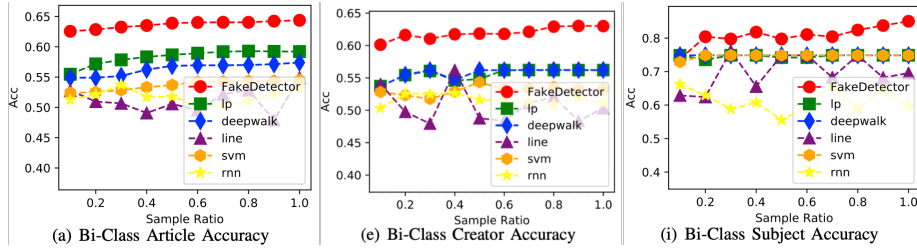


Figure 3.6: Accuracy Scores
[38]

The results show that the subject feature provides more information than any other feature and as a result, the accuracy has the highest value.

A failed attempt of implementing this algorithm has been tried, here are explained the issues faced while implementing and why the implementation failed. They state that all the data has been scrapped from the Politifact[1] website and that over 14000 fact checks have been retrieved. I have reproduced the scrapper two years later after the publication of this paper, a total of over 18000 fact checks. This dataset will be used in further sections of this project. The first issue is present in the body of the news. Politifact[1] does not provide the body of the news itself, instead, it provides the justification on why the volunteers believe that the given claim is false or true. It is common in almost every “body” of the news to have ending sentences


similar to this example “*Without substantial evidence for the data, we rate Harris’ claim False.*”[27]

Thus the body provides clear evidence of its label and it should not be used in the model. Politifact[1] does provide a source of the claim but it is not always located in the HTML container and thus there is no such method to consistently retrieve the original source. Furthermore, as seen in future sections, most of the claims do not originate from a textual source, other channels such as oral communications, videos, or images are very frequent in the dataset. It is known how this non-textual data has been treated or converted to textual information.

The architecture of the model also presents some impressions. As it can be seen in the diagram 3.5 the output of GDU cells is used as input of other GDU cells, the initialization of the model is not specified in the paper. It is completely unclear how to initialize the model without imposing all GDU outputs vectors as 0. On top of that, in the diagram, different HFLU and GDU cells are used for each of the features. It is unknown why there are 4 different news bodies and only 3 different cells for authors and subjects. Also, it is not clear what the sub-index in the diagram refers to. It can be assumed that each sub-index refers to different news or fact-checked claims in the dataset but this assumption fails as independent news will be mixed and thus the classification will be executed over mixed data entries.

The metrics results also show uncertainty. In the subsequent section of this project, the dataset used in this paper has been studied. In particular, a disparity assessment subjects feature reveals that 60% of the subjects appear only once in the dataset. This fact contrasts with the accuracy results of the model where the highest accuracy by far is found in the subject classification.

Code Reproduction

 GitHub Code [12]

A failed attempt to reproduce the code proposed is also part of this final bachelor project. The reason behind this failure is the fact that the article is imprecise in the definition of the architecture of the model, as argued in the Discussion section. The complexity of the connectivity together with the vague design specifications of the architecture and the fact the proper functionality of the model relies heavily on the connectivity lead to incompleting work.

I want to stress the idea that the number of assumptions needed to complete the code were too numerous. Even though the work could have been finished it was very obvious that the final model would have a poor performance and that little margin of improvement was left.

A huge effort has been put to contact the authors to ask for clarifications but all the attempts failed as I did not obtain any reply.

Nevertheless, this failed attempt lead to the development of my own model which uses the same dataset as the one proposed in the article. Surprisingly, the accuracy between these two models is extremely similar.

I decided to include incompleted work as I believe that the main goal of a final bachelor project is to demonstrate that the knowledge gained during the course of four years has been retained by the student. This work shows my programming and modeling skills and the ability to solve problems in an agile matter.

3.1.4 dEFEND: A System for Explainable Fake News Detection

Limeng Cui, Kai Shu, Suhang Wang, Dongwon Lee, Huan Liu [9]

Focuses on *explanatory* results. Uses comments on the news to detect which comments may justify the veracity or falsely of the claim as using them as the explanatory justification of the classification given by the model.

It is based on two main components, a web-based interface and the back-end consisting of a database with pre-trained results and a crawler to extract unseen news and its comments.

Interactive Backend

Two main modules:

- Detection Module

Generates a check-worthy score between 0 to 1 by using news content and user's comments.

- Attention Map Module

Outputs attention weights of each sentence and comments which show correlation to the major claim in the news, thus explainability. Uses Hoaxy API ¹ to get *diffusion paths* and Google News API ² to get the trending news and top claims.

¹Hoaxy: <https://hoaxy.osome.iu.edu/>

²Google News: <https://news.google.com>

Explainable Detection

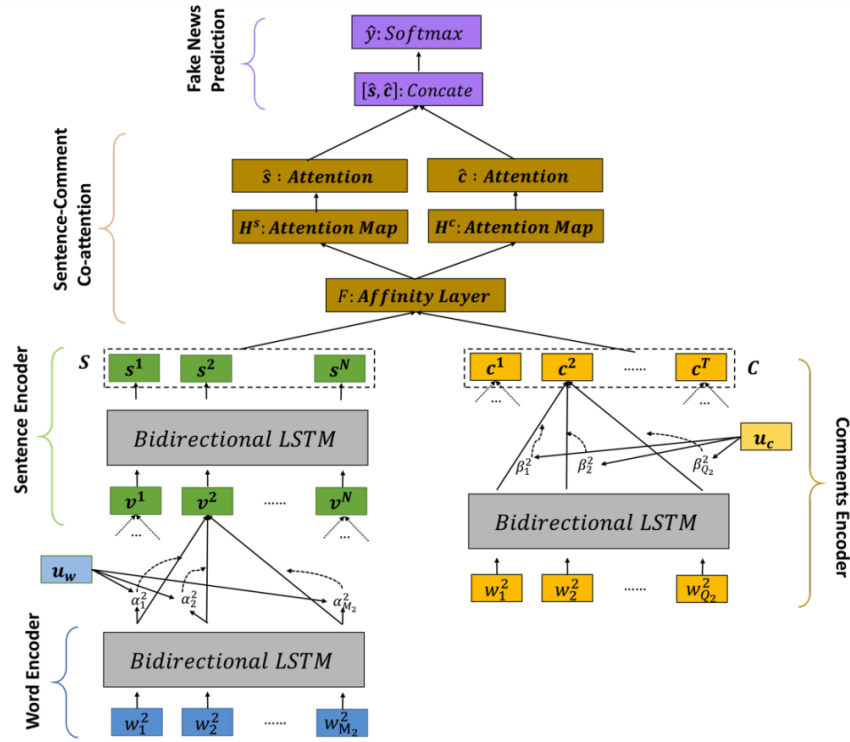


Figure 3.7: dEFEND Network Diagram
[9]

- News Content Encoding

It contains two sub-layers, first the *word encoder* followed by the *sentence encoder*.

- Word Encoder

Word encoder is based on a Bidirectional RNN using GRU. Bidirectionality implies that the given input sentence is passed from word w_0 to w_t and then from word w_t to w_0 . Then two outputs are generated:

$$\vec{h}_t^i = \overrightarrow{GRU}(w_t^i), t \in \{1, \dots, M_i\}$$

$$\overleftarrow{h}_t^i = \overleftarrow{GRU}(w_t^i), t \in \{M_i, \dots, 1\}$$

Then each word is encoded as the concatenation of the two GRU outputs. Then an *attention layer* is used to weigh the importance of words.

$$v^i = \sum_{t=1}^{M_i} \alpha_t^i h_t^i$$

– Sentence Encoder

Uses as input the output of the word encoder. It follows the same Bidirectional GRU architecture, $s_i \in 2d \times 1$. The output is the concatenation of the forward hidden state $\overrightarrow{h^i}$ and the backward hidden state $\overleftarrow{h^i}$.

- User Comments Encoding

Exact same mechanism used for word encoder for news content but applied to user comments on social media in news posts.

- Sentence-comment Co-attention

This layer tries to weigh the importance of both sentences in fake news as well as comments in social media posts. For the part of the sentence, as commented in other papers, most fake news are form mainly by true sentences that reinforce the fewer false sentences. At the same time, not all comments in the fake news post are correlated with the fake news itself, or in other words, do not provide a justification for why the fake news is false.

Two input matrices features are fed, the sentence matrix $S = [s^1, \dots, s^n] \in 2d \times N$ and the comments matrix $C = [c^1, \dots, c^T] \in 2d \times T$. This *affinity layer* has the following form:

$$F = \tanh(C^T W_l S)$$

Where $W_l \in 2d \times 2d$ and F is the *affinity matrix*.

The *affinity matrix* can be used as an input for the subsequent attention layers to learn to predict sentence and comment attention maps.

$$H^s = \tanh(W_s S + (W_c C) F), \quad H^c = \tanh(W_c C + (W_s S) F^T)$$

Where $W_s, W_c \in k \times 2d$ are the weight parameters.

Then the last layer makes a prediction over veracity.

$$a^s = \text{softmax}(w_{hs}^T H^s), \quad a^c = \text{softmax}(w_{hc}^T H^c)$$

Where $a^s \in 1 \times N$, $a^c \in 1 \times T$ are the attention probabilities of each sentence s^i and comment c^j , respectively.

- **Explainable Fake News Detection**

The above outputs a^s , a^c are concatenated and passed to a *softmax* layer. The outputs are the probabilities that the news is real and fake. Thus the minimization is over a *cross-entropy loss*.

Dataset

The dataset used is *FakeNewsNet: A Data Repository with News Content, Social Context and Spatialtemporal Information for Studying Fake News on Social Media*[31] which contains data from Politifact[1] and Gossip Cop³. The features are ID, `news_urls`, `title`, `tweet_id`.

Discussion

The paper shows that element that are external from a news provide more useful information than the news itself. Fake news are usually written in a misleading manner to fool the reader in to believing a misinformation, one of the strategies to achieve it is to either write small sentences with direct false information or long news articles containing mostly true sentences to cover the fewer false sentences. Using NLP over news textual information is a naive approach since false news would only be detected if their writing style differs over true news consistently or there are contradictions within the text. The best approach is the search for a justification that proves the veracity of the news. In this scientific work the commentaries on news social media posts are used as guidance for the classification and also provide a key aspect of this paper *explainability*.

Explainability in ML/DL models accounts for the capacity of the system to explain the outputs. In this particular case, explainability concerns the capacity that the algorithm has to justify why particular news is believed to be false or true. This is achieved by displaying the comments in social media posts that the algorithm has weighted the greatest *affinity* to the news.

³Gossip Cop: <https://www.gossipcop.com/>

The implementation of this algorithm is completely feasible as the dataset is accessible and the computational complexity of the model can be executed in commodity hardware. The implementation itself can be cumbersome since there is not much detailed specification in many of the crucial aspects of the model. For example, it is unknown how the words are encoded, there are experiments neither no metrics results that explain the performance of the algorithm.

The model can be generalized easily and can be applied to any given news easily. Nevertheless, there are some issues with this solution. First, the used data is really well-curated and the IDs of the tweets are selected such that the justification over false news is found within this list. In a real-world application, the number of responses to the news posts would be much larger, it is unknown how this fact would affect the performance of the algorithm. Moreover, it is required that the news social receives interaction before the classification, this is a drawback since the ultimate goal is to prevent the broadcasting of misinformation in an early stage.

Code Reproduction

 [GitHub Code \[10\]](#)

As part of this final project, the code proposed in this paper has been reproduced which can be found in the GitHub link.

The model has been programmed in Python using the library **TensorFlow** as well as **Keras** library which is developed on top of **TensorFlow**.

The dataset used is the same as the one proposed in the article [31]. The dataset found in [31] does not contain the textual information of the news article and the tweets instead it contains the hyperlinks to the news articles webpage and the IDs of the tweets. To easy the obtaining of the dataset the code posted in Harvard Dataverse [34] has been used to generate the dataset containing the textual information.

Chapter 4

FAKE NEWS DETECTOR

This section is focused on the implementation aspects of this final project. It is divided into Iterations. The set of iterations are ordered chronologically. The purpose of this organization is to capture the evolution of the project and the logical reasoning behind each decision and to justify the subsequent steps in the implementation. Each iteration finishes with a conclusion which wraps up the work in that particular iteration and motivates the upcoming iteration.

4.1 Iteration 1 - Dataset

Most of the papers refer to the Politifact[1] website as their source for the creation of their datasets. As part of this project, a Scraper has been built to retrieve all the fact checks present on the Politifact website. An extensive Jupyter Notebook defines the process and required scripts to execute this task.

4.1.1 Scraper

 [GitHub Code \[16\]](#)

The scraping process is divided into two subroutines. Both implemented in Python 3.

1. The **first subroutine** consists of retrieving from the list page all the URLs which contain the fact checks. To do so three main libraries have

been used, Selenium¹, BeautifulSoup², and Request³. Request has been used to download the HTML file then BeautifulSoup has been used to parse the HTML and retrieve all links. To navigate through all the list pages, Selenium has been used to click the “Next” button at the end of each listing page. All the URLs have been saved in a *.csv* file to be employed by the next subroutine.

2. The **second subroutine** iterates over the list of URLs. Similarly as before Request library assisted on the task of downloading the HTML file and BeautifulSoup parsed the HTML and selected the required data fields.

In each of the fact checks, we can retrieve the following elements which characterize the news or claim.

- **Title:** Contains a brief description of the news or claim. E.g, “*As of today, no one has the right to film or photograph Mr. Biden as he climbs the flight of stairs. New White House spokesman orders!!?*” [26]
- **Author:** Entity or individual person to whom the fact check is referring. Source of the news. E.g “*TikTok Post*”. [26].
- **Tags:** Label characterizing the news or claim, describes the topics related to it. E.g “*National*”, “*Joe Biden*”. [26]
- **Body:** Extension of the Title. Contains an in-depth description of the claim by the individual or entity and the justification for the given rating. The body presents a big issue when used in an ML/DL model since it contains phrases that make explicit reference to the given rating by the website. For example “*we conclude the claim is false*”, “*we rate this claim as mostly false*”, etc. Thus when used in an NLP model those hints will create false satisfactory results since the model will learn to identify the hints and use them as their classification criteria. The ultimate goal is to classify raw news not to classify already classified news. That is why this element has not been used in the consecutive implemented models. E.g, “*A TikTok video claims there’s a new rule:*

¹Selenium: <https://www.selenium.dev/>

²BeautifulSoup: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

³Request: <https://docs.python-requests.org/en/master/>

No cameras allowed when President Joe Biden climbs the stairs to Air Force One...with many indicating they believed the claim to be true. It is not.” [26]

- **Rating:** Politifact[1] uses a ranked grading system that rates news from extremely false claims to completely true claims. The rating is divided into six different categories from true to false; “*True*”, “*Mostly True*”, “*Half True*”, “*Mostly False*”, “*False*”, “*Pants-on-fire*”. Other ratings also appear in a small subset of the reviews. Those ratings do not reference the veracity of the claim instead rate the change of positioning by an organization or individual. The labels used from this kind of rating are “*No flip*”, “*Half flip*”, “*Full-flop*”.
- **Sources:** Links to the origin of the claim. This element has not been used since there is no consistency in the location of the sources. It is impossible for an automated system to consistently retrieve the source. Moreover, some of the sources are digital videos or images which are out of the scope of the model.

4.1.2 Exploratory Data Analysis

 GitHub Code [11]

The goal of this section is to apply EDA techniques such as statistical measurements and visualization methods to understand and characterize the scrapper dataset. As part of this study, it will be assessed if the data can be used by the subsequent classification task. Other aspects such as the political bias in the fact checks will be reviewed from a completely impartial point of view.

The final dataset has the following form:

Title	Tags	Author	Rating
Says Ron Johnson...	Corporations,Small...	Russ Feingold	half-true
”Forty percent of...	Immigration,National...,	Steve Case	half-true
On considering U.S...	Bipartisanship,S...	Tammy Baldwin	half-flip
”United States of...	New Jersey...	Jim Whelan	mostly-true
Says Virginia Gov...	Fake news,Facebook...	Facebook posts	pants-fire

Table 4.1: Politifact Scrapped Dataset

It contains a total of 18066 records this number will get reduced as the dataset gets curated.

Ratings

The first step is to analyze all the unique values that constitute the Rating column. As anticipated before, some ratings do not make explicit reference to a fact check, thus all the data entries with those values are deleted. In the plot below we can see a graphical representation of the rating frequencies in the dataset.

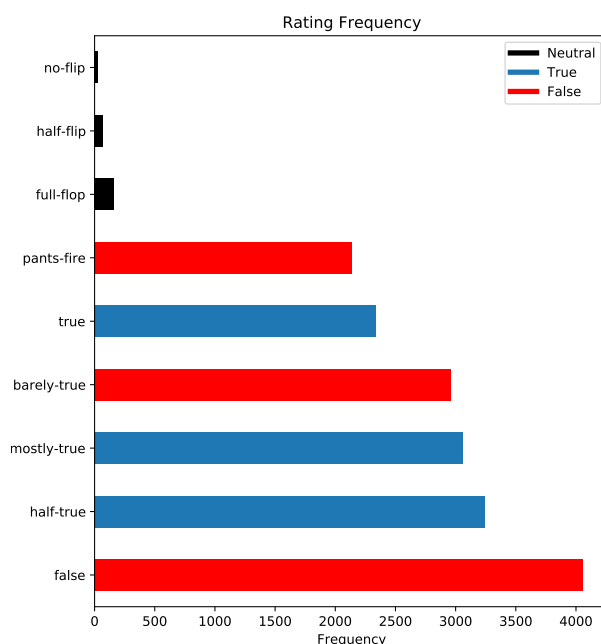


Figure 4.1: Rating Labels Frequency

It can be appreciated that there is an acceptable balancing between False and True news.

Authors

A simple visualization of the top ten most reviewed authors.

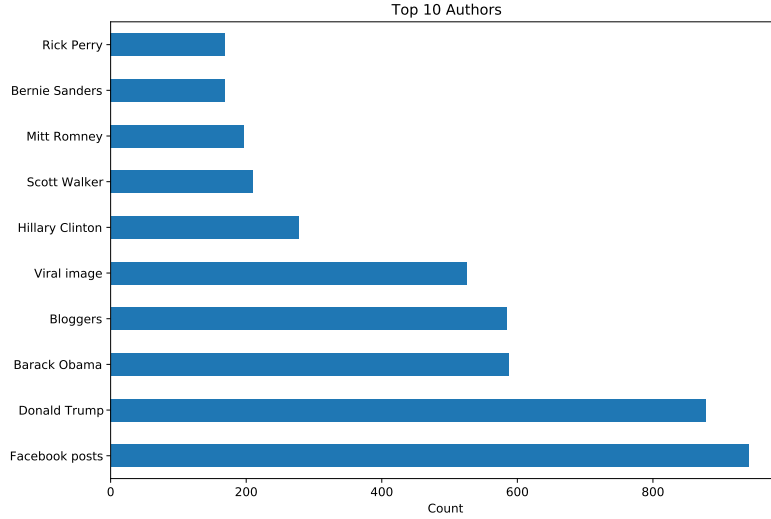


Figure 4.2: Top 10 most Frequent Authors

The dataset can be divided into two subsets, one which contains only true news and one that contains false news. Then we evaluate the **trustworthiness** rate TR of each Author by applying the following metric:

$$TR = \frac{TN}{TN + FN}$$

Where TN is the count of true news which that particular author has and FN is the number of false news which that particular author has. We can compute the opposite metric **untrustworthiness** rate UTR as follows:

$$UTR = \frac{FN}{TN + FN} = 1 - TR$$

The following pie plot shows for each of the top 10 most reviewed authors their respective TR and UTR score, and the total number of reviews.

The following plots show the top ten authors with at least 30 appearances in the dataset with the highest TR and UTR score.

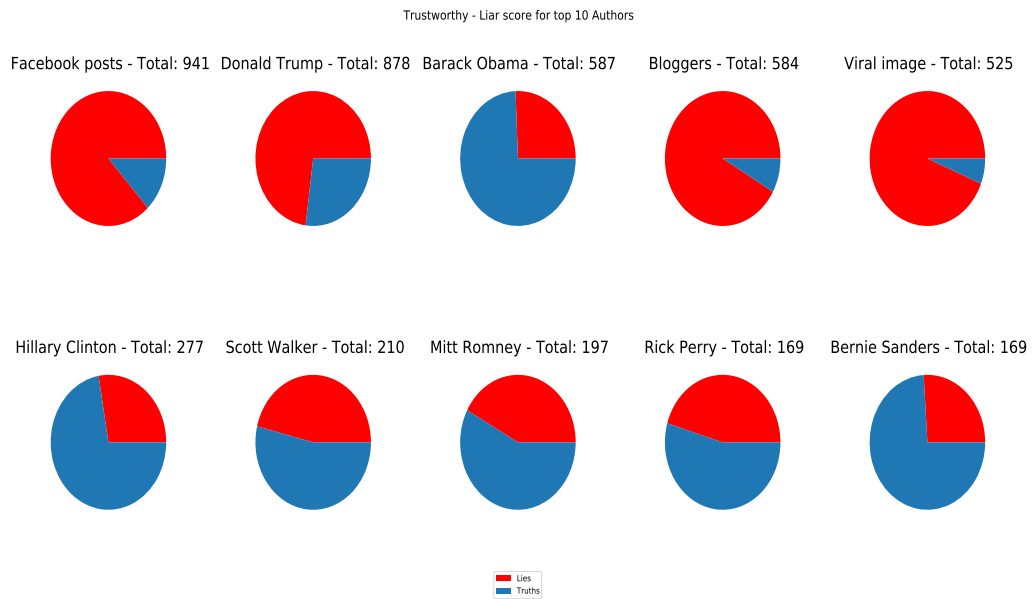


Figure 4.3: Pie Plot Top 10 most frequent Authors and un/trustworthiness score

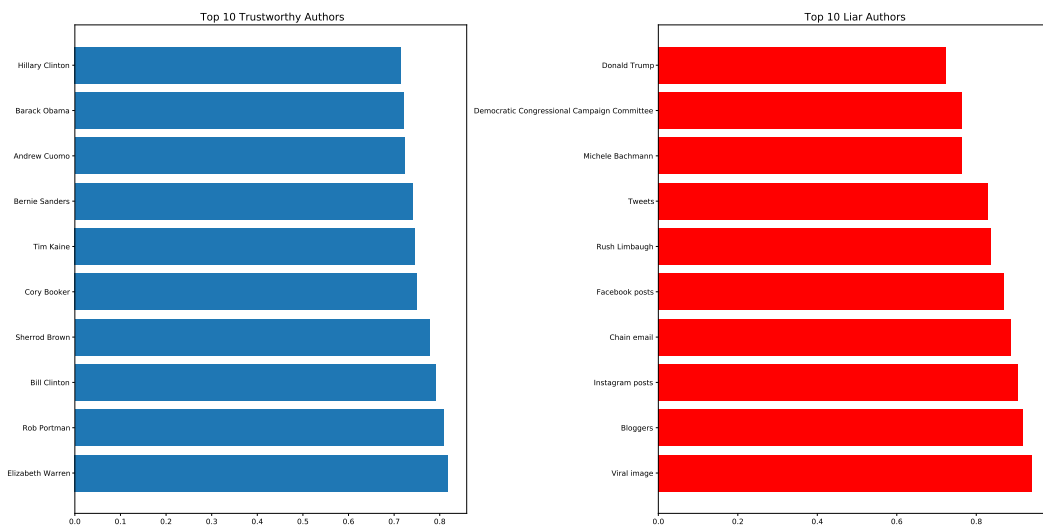


Figure 4.4: Top 10 Author by TR and UTR scores.

Note that some authors are not natural persons. When news gets viral on the internet and an author is an unknown person that news gets the authority of the data type or the transmission channel, such as Viral Image or Twitter.

Classification Model Implications

There are a total of 4253 unique authors. This is a concerning aspect since there is too much disparity compared to the size of the dataset. There are a total of 2616 authors that appear only once which represents a total of 61.5% of the total number of authors. The information provided by this feature for these authors is null. In the learning phase, these authors will only introduce noise. In the validation or testing phase, it will be the first time that the model sees these authors thus it would not have previous knowledge to assess the classification. Nevertheless, since more features will be used in the classifier these data records will not be deleted in this section.

Bias assessment of the dataset

For the 10 authors in the most trustworthy plot, there are 9 which are affiliated with the Democrat party, and 1 which are affiliated with the Republican party. On the other hand, for the 10 authors in the most untrustworthy plot, there are 1 which are affiliated with the Democrat party, and 3 which are affiliated with the Republican party. It is important to mention that in this plot only 4 authors could be assigned to a political party.

Title

To analyze this paper the library `scattertext` [21] has been used. The final result of this subsection is a scatter plot where a point is a word. Each axis is associated with a category, in this case, True and False. The closer a word it is to the y-axis or x-axis the more related is to the associated category. Height indicates a high frequency of appearances in documents labeled with the y-axis category. The further right a point appears, the more its corresponding word has been used in documents labeled with the category associated with the x-axis. Interesting words are those that appear either in the top left corner or in the bottom right corner as those terms have a high correlation with the corresponding category.

To achieve the final result the following steps have been taken:

1. A new column has been added to the dataset name Label which can take either True or False. If the Rating column has one of the following values: 'true', 'mostly-true', or 'half-true' Label takes value True.

Similarly, if the Rating column has one of the following values: 'false', 'barely-true' or pants-fire' Label takes value False.


2. To reduce the diversity of words NLP preprocessing techniques have been applied to transform the original Titles. The transformation applied are, lowering all capital letters, removing punctuation marks and, removing common terms with the help of the library NLTK⁴ by importing stopwords dictionary and removing, from the Titles, the words appearing in the dictionary. Finally, all titles have been parsed using Spacy⁵ and NLP function, this step is a requirement for `scattertext`⁶.

Example of preprocessing:

- Original title: *"Says Ron Johnson referred to "The Lego Movie" as an "insidious anti-business conspiracy." "*
- Preprocessing output: "says ron johnson referred lego movie insidious antibusiness conspiracy"

3. Lastly, the scatter plot is generated with the required `scattertext` functions.

The coordinates of each word are determined by the probability of a particular word appearing in one of the two categories, true and false. As explained in the paper *Scattertext: a "Browser-Based Tool for Visualizing how Corpora Differ"*[21].

The resulting plot is saved in an HTML file. I highly encourage the reader to open the file, *Titlescatterplot.html* , and to use the interactive features for analyzing the content of the titles.

⁴NLTK: <https://www.nltk.org/>

⁵Spacy: <https://spacy.io/>

⁶scattertext: <https://github.com/JasonKessler/scattertext>

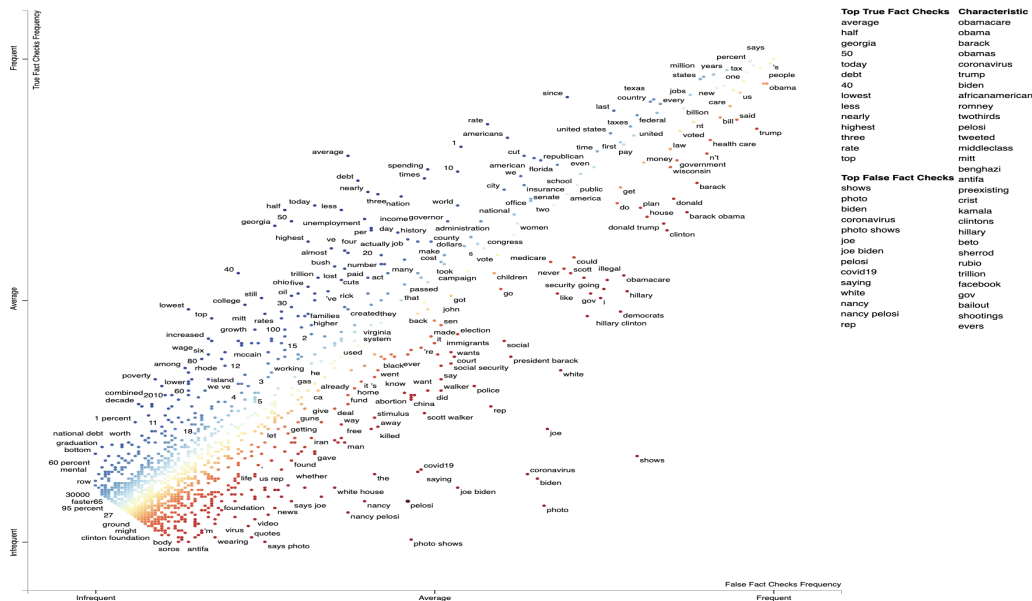


Figure 4.5: Title Scatter Plot of Terms

In the scatter plot above the x-axis is associated with False news while the y-axis is associated with True news. Then terms appearing in the top left corner are highly correlated with True news while terms appearing in the bottom right corner are highly correlated with False news.

Classification Model Implications

As stated earlier, words that appear in either the top left corner or in the bottom right corner are words of interest. When the classifier ‘sees’ those words in particular news it will be a clear indicator that the news is either False or True. Unfortunately, no such terms appear in the plot. The models presented in upcoming sections of this project use complex cells such as LSTM which are capable of capturing long-term dependencies in a sequence of data. In theory, the fact that no words appear in previously defined interesting regions should not have direct negative effects on the ability of LSTM cells to evaluate the associated label. It would use the intricate relationships between terms instead of only using the atomic information that a single term gives. However, the average number of words in Titles is 18 an unsettling fact since LSTM cells may fail to capture relationships between terms. Further discussions about Titles arrive in future sections

Bias assessment of the dataset

Below are listed the most correlated terms for True and False news.

- True news: average, half, Georgia, 50, today, debt, 40, lowest, less, nearly, highest, three, rate, top.
- False news: shows, photo, Biden, coronavirus, photo, shows, Joe, Joe Biden, Pelosi, covid19, saying, white, Nancy, Nancy Pelosi, rep.

For true news, no interesting words appear that indicate any kind of political bias as none of the words make strict reference to a person which a political opinion could be associated with. In opposition, false list do contain words associated with natural persons as well as trending topics. “*Shows*” and “*photo*” are one of the most correlated words to fake news that is directly related to 4.4 as the fakest author is indeed viral images. There are a total of 103 false news titles which contain the bi-gram “*photo shows*” while only 2 true news titles contain this bi-gram.

Three n-grams refer to the current president of the United States of America Joe Biden. Although Joe Biden appears in the false news list out of the 154 news which contains the bi-gram “Joe Biden” only 1 of them has as author Joe Biden. This implies that all of these 154 false news are defamations of Joe Biden and that Politifact[1] is denying the false claims against him. A similar situation happens with the Democrat speaker of the United States House of Representatives, Nancy Pelosi. Out of the 101 mentions of Nancy Pelosi in false news, none of them has as author Nancy Pelosi. This implies that all of this 101 false news are direct defamations to Nancy Peloci and that Politifact is denying these false claims. Note that this situation happened twice for the Democrat party and zero for the Republican party.

Tags

By dividing the dataset into two subsets True and False we can get the most common tags per subset. The first plot illustrates the top 20 tags used in True news and the respective frequency for both True and False news. Likewise, the second plot shows the top 20 tags used in False news.

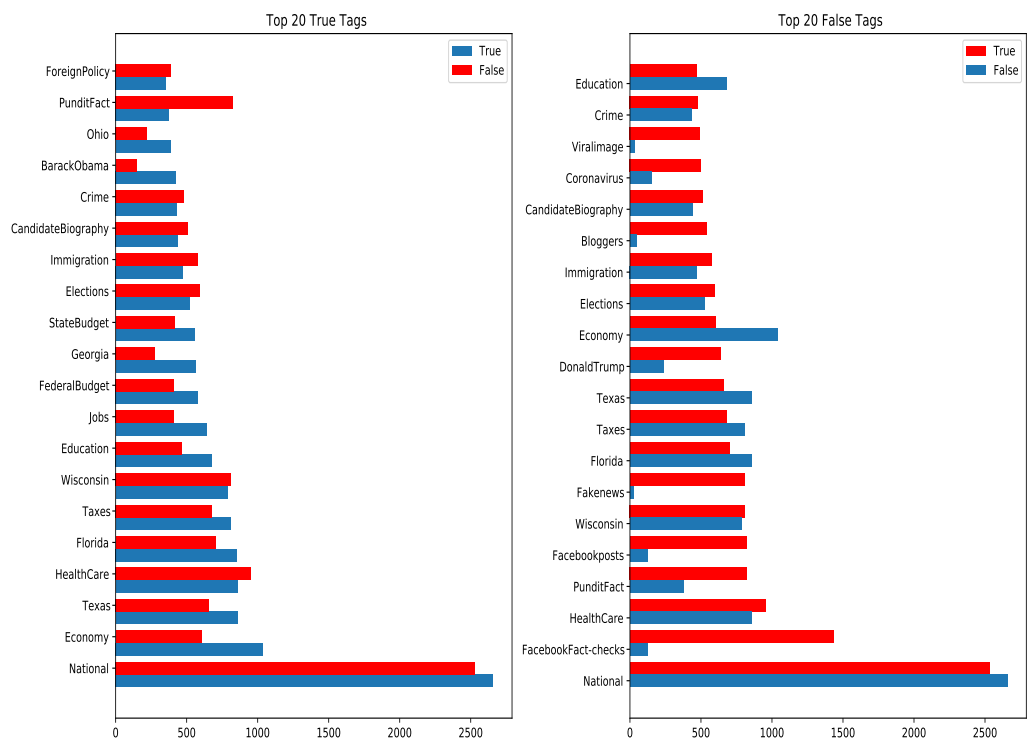


Figure 4.6: Most frequent True/False

Combining Authors with Tags the following bar plot displays the top 10 used tags per the top 10 most frequent authors.

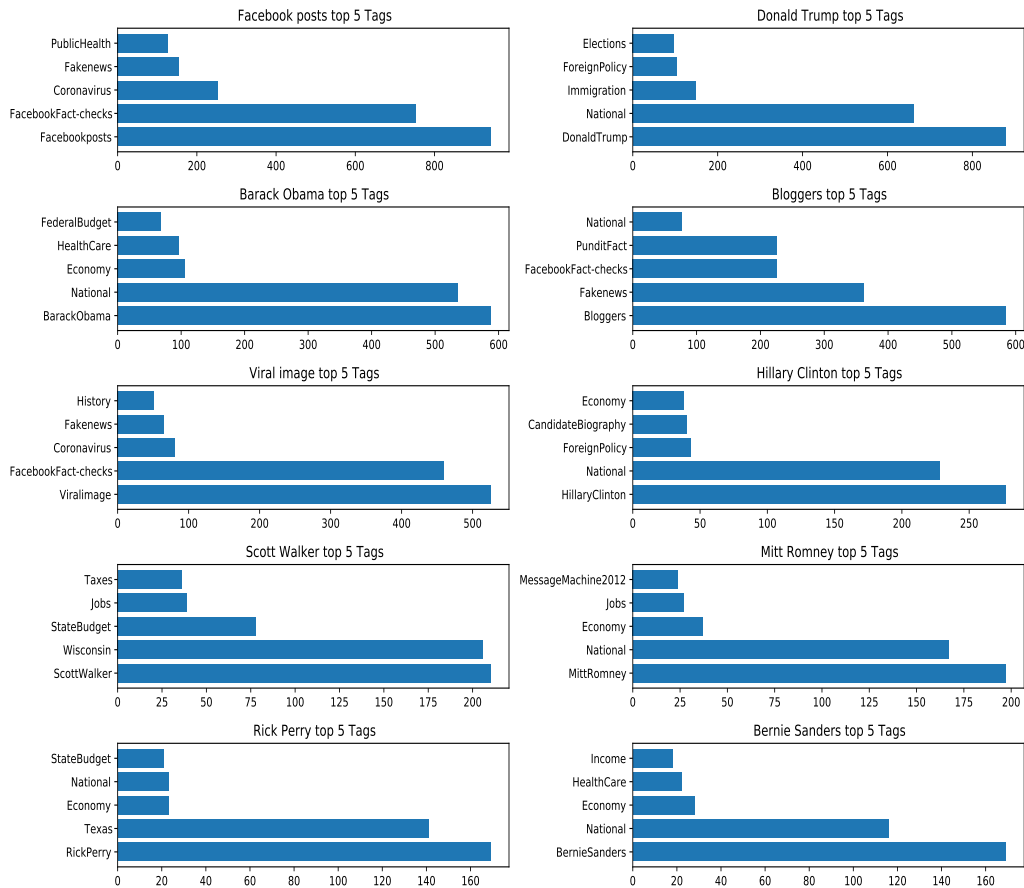


Figure 4.7: Most frequent tags per most frequent authors

Three word-cloud visualizations are available in the Append section, see figure... Figure 1 displays the most used tags. Fig 2 shows the most used tags in True news. Fig 3 shows the most used tags in False News.

Classification Model Implications

Analyzing the disparity in tags is a good assessment of how much information tags are going to provide to the classification model. A high disparity will be an indicator of noisy data and thus a bad classification performance. A low disparity will be an indicator of consistent data which may be an indicator of the good performance of the algorithm.

There is a total of 4428 unique tags in the dataset. The table below shows the number of tags appearing a particular number of times in the dataset and the percentage that those tags represent over the total number of unique tags.

Frequency	Number of Tags	Percentage over Total
1	2617	59.10%
2	661	14.93%
3	280	6.32%

Table 4.2: Tags Frequency - Number of Tags - Percentage

These results are clear indicators of the huge disparity in Tags as almost 60% of the tags appear only once in more than 18000 data entries. On average each news has a total of 4 different tags with a standard deviation of 1.2 tags. Then it is assumed that will not provide useful information to the classifier since the feature tags are extremely noisy.

Moreover in 4.6 it can be appreciated that almost none of the tags has a massive representation in either True or False news but minor representation in the opposite category. This has the implication that most of the tags will provide unimportant information to the classifier.

4.1.3 Conclusions

Scraper

The data has been properly scrapped as no **Nan** values appear in the dataset and all data features are consistent with their content.

Bias assessment of the dataset

There is some clear political bias on the dataset as the distribution of false news is uneven between political parties. In particular, Democrats representatives in the dataset appear on average as authors of true news while Republicans representatives appear on average as authors of false news. Moreover, there are noticeable efforts on denying false claims and hoaxes over Democrat party affiliates as concluded in the Title section.

Classification Model Implications

There are visible pieces of evidence in many of the features which show poor data consistency. Both Authors and Tags features show an incredibly noisy pattern which has a direct implication for the classifier as it will fail to capture patterns linked to false or true news. The Title feature has a small length of words which may make the algorithm fail to capture the intricate relations between words that codify the truthiness or falseness of news. Since a small number of words appear in the title the algorithm could learn to detect highly correlated words to one particular label and use those to guide the classification criteria. Unfortunately, there are no such terms in the scatter plot 4.5. Thus it can be concluded that there are low expectations in the performance of the classifier models.

4.2 Iteration 2 - Proposed Model

The development of this model has been incremental. Meaning that small changes and improvements have been added one by one. The goal of this kind of implementation is to build a robust final model based on all the trials and errors.

Here are explained all implementation details common to all the different versions of the model to avoid repetition.

Modeling Framework

The problem that the model will solve is a classification problem. Given a set of features and target labels the model will use the features as a guide to develop a classification criterion that maps a pattern in the input features to an output label. In this particular case, the set of input features will be the titles, the authors, and the tags, and the target label will be the simplified rating of either true or false.

- **Textual Variables**

Title variable is a textual data type. As explained in refBackground chapter textual data is treated in ML/DL models using a set of techniques known as NLP or Natural Language Processing. Thus in the different implementations, a common element would be the use of either LSTM cells or GRU cells to capture the significance codified in the collection of terms used in each title.

- **Categorical Variables**

Tags and authors will be treated as categorical variables. A categorical variable takes one or more values of a set of possible values. The space of values is defined by the set of authors and tags and each news can take one author and an undefined number of tags.

- **Model Framework**

Given the nature of the titles feature a Deep Learning model framework will be the construction base of our model.

- **Programming Framework**

All the models will be programmed in Python 3 taking a particular scripting form known as Notebooks. Python has been the standard language for programming for writing ML/DL models in the last few years as its simple and consistent. It includes a large set of libraries and frameworks which eases the work by focusing on the mathematical aspects of the implementation. Two libraries are dominating the Deep Learning field, **TensorFlow** ⁷ and **PyTorch** ⁸. Both frameworks will be used to enhance the learning aspects of the project.

⁷TensorFlow: <https://www.tensorflow.org/>

⁸PyTorch: <https://pytorch.org/>

4.2.1 Version 1 - LSTM over Titles

 [GitHub Code \[13\]](#)

In this first implementation, only the feature Title has been used to define the classification criteria. Here is a diagram of the network.

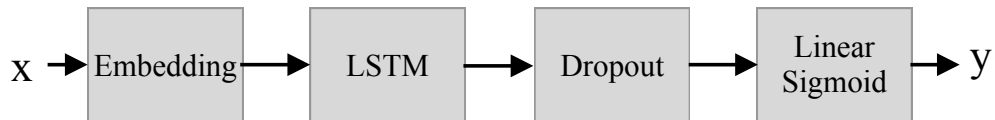


Figure 4.8: Version 1 Diagram

Discussion

Metrics reveal that the model completely malfunctions.

Metric	Score
Accuracy	0.5140
Recall	1.0
Precision	0.5140
F1	0.6790

Table 4.3: Version 1 Metrics

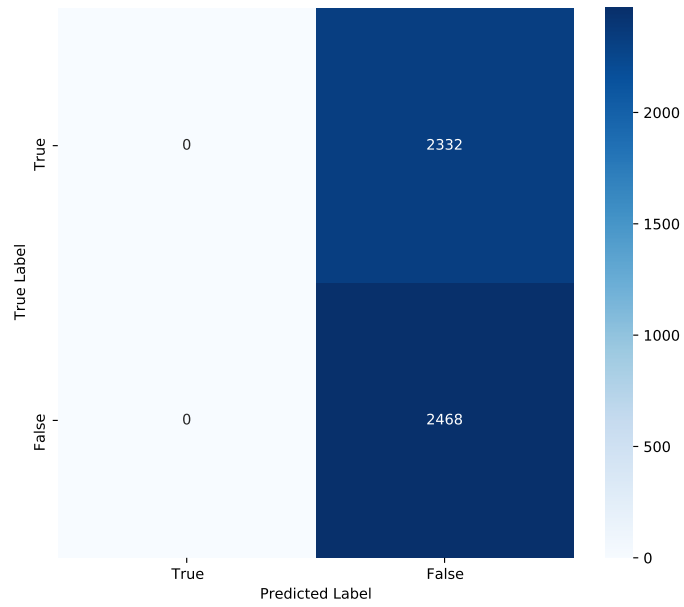


Figure 4.9: Confusion Matrix Version 1

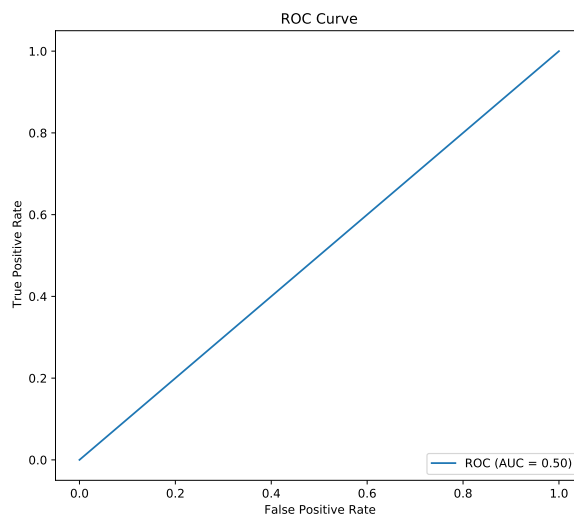


Figure 4.10: ROC Curve Version 1

All these metrics indicate that the Title feature does not codify relevant information regarding the veracity of a news item. Moreover, it can be seen that in fact, the model has decided to mark all news items as false. By looking at the outputs predictions it can be seen that all news items have received the same value. This is a confirmation of the conclusion in section Exploratory Data Analysis Title subsection 4.1.2. By expecting some of the titles it can be understood why the classifier fails in its task.

“Says Ron Johnson referred to ”The Lego Movie” as an insidious anti-business conspiracy.”

“Forty percent of the Fortune 500 were started either by immigrants or children of immigrants.”

“United States of America is twenty-sixth in school performance in the world.”

The length of titles is too small to codify underlying information that could be associated with one of two labels. The reader may also notice that for him it is impossible to determine the veracity of any of the given titles. Similarly, as the reader, the algorithm has decided to have rigid criteria. Since it can not develop criteria for guiding the decision over the textual information the model simply assigns all news to false as it is the label that minimizes the loss function.

To improve the performance of the algorithm an alternative model using pre-trained embedding has been implemented. Unfortunately, no improvement has been detected. One of the reasons for this lack of improvement may be related to the fact that more 28% of the words do not appear in the pre-trained embedding dictionary. The motivation for this variant is to ease the model numerical complexity by substituting the trainable embedding layer for an already trained embedding.

4.2.2 Version 2 - Authors

 [GitHub Code \[14\]](#)

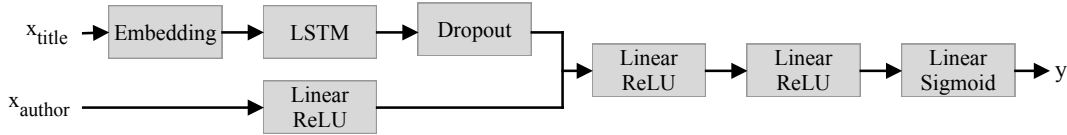


Figure 4.11: Version 2 Diagram

The author feature is a categorical data type. Categorical data is usually easy to treat implying that simple models should be able to adapt easily capturing the patterns and correlations codified in the input dataset.

Preprocessing

Categorical data is best encoded using one-hot encoding. To achieve one-hot encoding the following steps have been defined. A list of all unique authors is generated from the dataset. Then for each author, an index is mapped, e.g. “Joe Biden”: 0, “Donald Trump”: 1,... For each entry in the dataset, the original string representing the author is then substituted by a vector v of dimension V , where V is the number of unique authors with a certain frequency. All elements in vector v are 0, except for the element in the index position mapped to the author’s string, which takes value 1.

As mentioned in the Exploratory Data Analysis 4.1.2 subsection there are more than 2500 authors that appear only once. Those authors will act as noise for the model as it would not be able to find a correlation between an author and a label. Thus, after long trial and error testing, all of the authors with an appearance frequency of less than 5 will be removed from the dataset. This decision is motivated by the increase in accuracy and general performance of the classifier.

Linear and ReLU

The two subsequent Linear ReLU layers have been added mainly to decrease the dimensionality progressively. The concatenation of the LSTM and the Author layers generates a tensor of size 9472 and the output vector is of size 1, directly compressing the vector from 9472 to 1 decreased the performance of the algorithm.

Discussion

Metric	Score
Accuracy	0.6847
Recall	0.6407
Precision	0.7390
F1	0.6864

Table 4.4: Version 2 Metrics

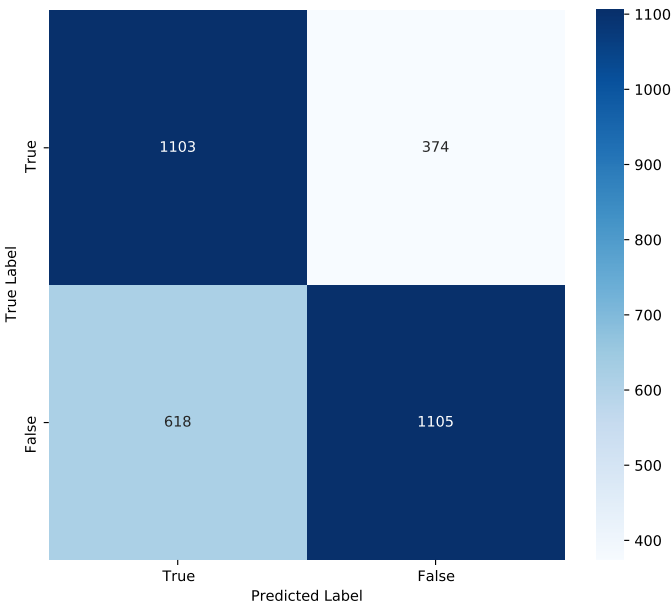


Figure 4.12: Confusion Matrix Version 2

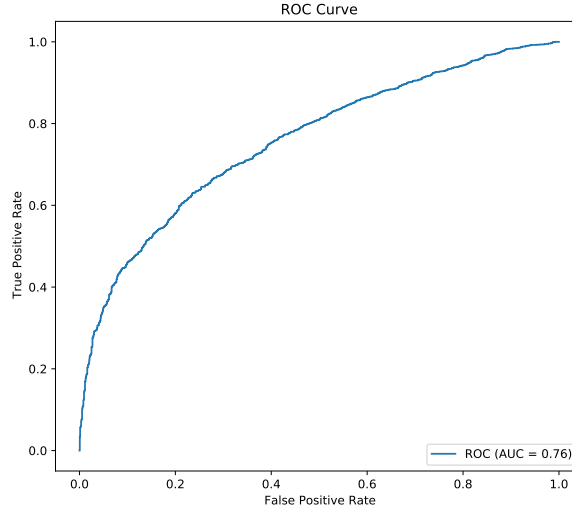


Figure 4.13: ROC Curve Version 2

The metrics present a clear improvement in the performance of the model. By comparing the Confusion Matrix between the previous implementation and the current one it can be seen that the algorithm has defined more complex criteria than just classifying everything as false. Thus it can be concluded that the authors feature encodes more information than the title itself.

In the EDA subsection 4.1.2 regarding authors feature a list of the top ten trustworthy authors and untrustworthy authors have been generated 4.4. To make a bias study of the dataset and to evaluate the performance of the classifier all the data entries with the author in either of the list have been passed to the model to study the results. As mentioned, 9 out of the 10 authors in the trustworthy list affiliate to the Democrat party and 1 to the Republican party. Differently, on the untrustworthy author's list, 1 is associated with the Democrat party, and 3 is associated with the Republican party. The remaining authors do not have an assignable political association.

- Top trustworthy authors: *'HillaryClinton', 'BarackObama', 'AndrewCuomo', 'BernieSanders', 'TimKaine', 'CoryBooker', 'SherrodBrown', 'BillClinton', 'RobPortman', 'ElizabethWarren'*.
- Top untrustworthy authors: *'DonaldTrump', 'DemocraticCongression-*

alCampaignCommittee', 'MicheleBachmann', 'Tweets', 'RushLimbaugh', 'Facebookposts', 'Chainemail', 'Instagramposts', 'Bloggers', 'Viralimage'.

Top trustworthy authors metrics

Metric	Score
Accuracy	0.7492
Recall	0.0556
Precision	0.5455
F1	0.1008

Table 4.5: Version 2 Trustworthy Authors Metrics

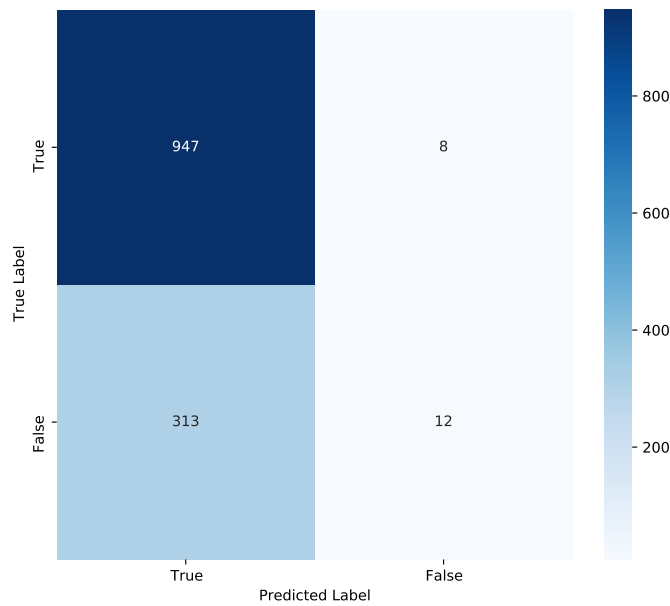


Figure 4.14: Confusion Matrix Version 2 True Authors

There is an improvement in the accuracy. This fact indicates that most of the other authors who have a small appearance frequency in the dataset introduce noise to the data and thus they decrease the accuracy of the model.

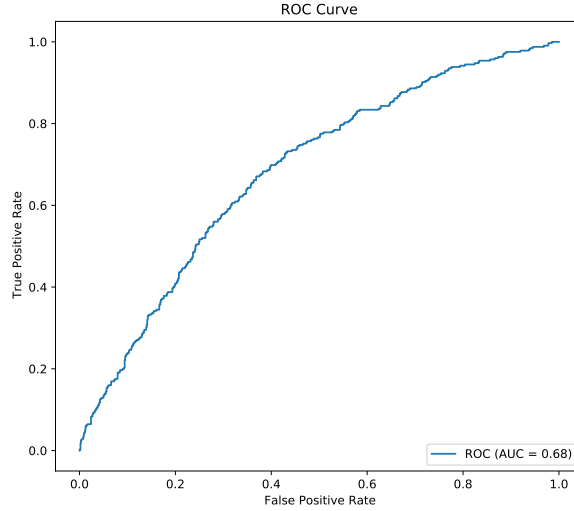


Figure 4.15: ROC Curve Version 2 True Authors

The model has an evident bias over these authors which implies that the model has learned the general rule detected in the EDA 4.1.2 subsection of being trustworthy. It is appreciated in both the precision and recall score. Since it tends to classify as true news the precision score increases and the recall score decreases. It is also important to mention that the number of False Positives is lower than the number of True Positives where being positive implies that the news is false. This is an important fact since for these authors the bold assumption by the algorithm should be to assume a negative classification as these authors are characterized for generally saying the truth. This phenomenon implies that the titles by their own, as seen in the previous section, do not codify meaningful information but titles synergize with authors generating meaningful information. Like so it can be justified why the algorithm has managed to output more True Positives than False Positives against the general rule.

Top untrustworthy authors metrics

When the algorithm classifies this subset of data the improvement is notorious. There is an obvious reason for that, as seen in the EDA section 4.1.2 these authors are characterized by being extremely unreliable sources of information. By looking at their respective untrustworthiness score it can be appreciated that the scores are much higher than the trustworthy authors.

Metric	Score
Accuracy	0.8564
Recall	0.9732
Precision	0.8724
F1	0.9200

Table 4.6: Version 2 Untrustworthy Authors Metrics

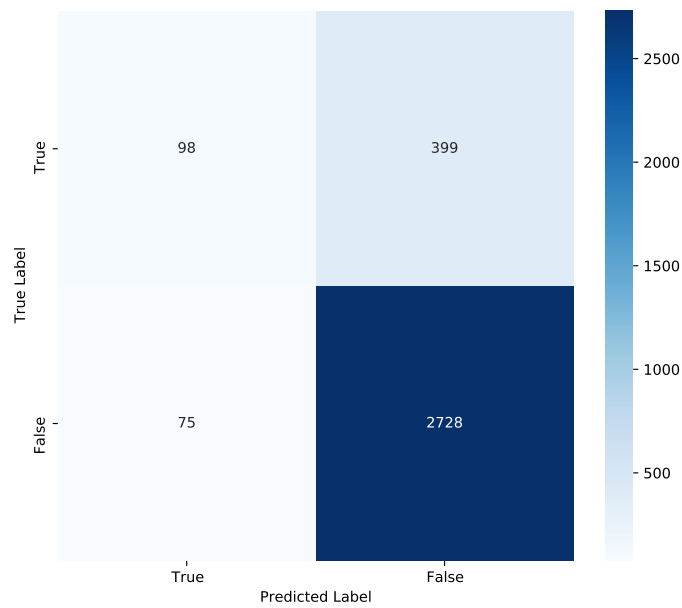


Figure 4.16: Confusion Matrix Version 2 False Authors

This implies that making a strict rule of always classifying these authors as false will always lead to a better accuracy compared to the other subgroup of trustworthy authors. On top of this, some authors in this list, for instance, viral image, Facebook post, or bloggers are at the same time the most frequent authors in the entire dataset, as seen in subsection EDA 4.1.2. These two factors cause an increase in accuracy.

Equally as before, there is a clear synergy between titles and authors. For this subgroup, the bold assumption that the algorithm could make is to classify all news as false. But it has a diverse classification having a greater

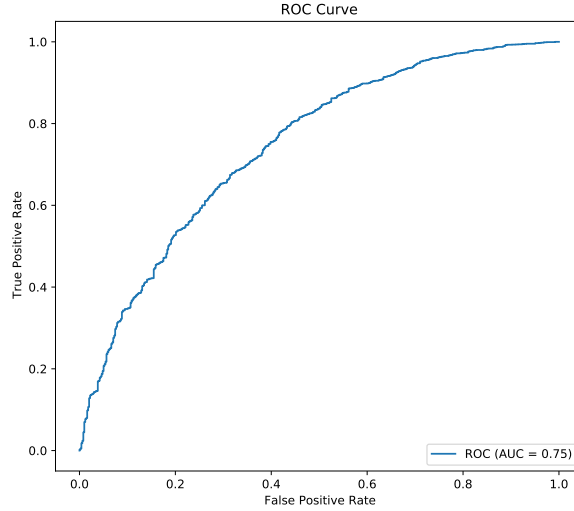


Figure 4.17: ROC Curve Version 2 False Authors

number of True Negative rate compared to the False Negative rate.

In general conclusion, it can be stated that the introduction of the author feature has had an impact on the classification. Now the algorithm has developed more intricate and diverse criteria and has shown that titles feature on its own does not codify meaningful information but synergizes with the introduced feature authors. A big part of the errors made by the model are caused by the noisiness of the data. When selecting specific subsets of the entire dataset, which are known to be less noisy, a huge improvement is appreciated.

4.2.3 Version 3 - Tags

 GitHub Code [15]

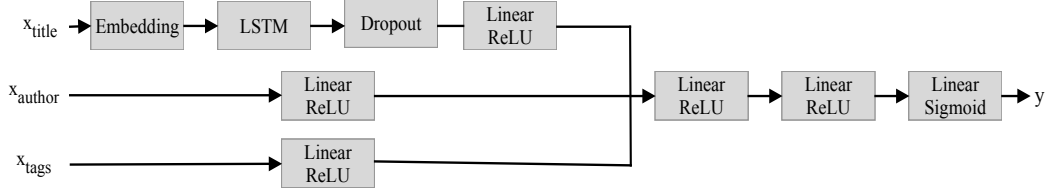


Figure 4.18: Version 3 Diagram

Similarly, as authors, tags are categorical data thus similar preprocessing methods and layers used for processing the information will be implemented. Also in this version, a linear layer followed by a ReLU activation function has been added to the model. The goal of this addition is to reduce the dimensionality of the LSTM layer from 8960 to 512. The motivation between dimensionality reduction is caused by the fact that a bigger vector may cause a higher impact on the decision criteria, taking into account that the title layer provides almost no information, reducing its dimensionality improves the performance.

Preprocessing

One-hot encoding is required to transform the strings into numerical data. Tags differ from authors as they do not only contain a single value for each news, e.g one news can have multiple tags such as *Corporations*, *Small-Business*, *Wisconsin*. The first step to generate the required encoding is to generate a vector of length T where T is the total number of tags that appear with a frequency greater than or equal to a certain frequency. Then for each of the remaining tags, a number is mapped, e.g *Corporations*: 0, *SmallBusiness*: 1,... For each of the news, a vector of size T is generated where. This vector contains 0 in all positions except for those indexes that correspond to the mapped number of each tag string appearing in that particular news. Thus, in general, most one-hot encoded tags contain several 1 in the vector.

Linear ReLU

The same reasoning as the author feature can be applied for tag features. Other activation functions have been tested but ReLU output the most ac-

curate model. Leaky ReLU also outputs a similar accuracy than ReLU but ReLU was slightly better.

Discussion

Tags feature does not provide meaningful information to the model. The minor improvements in the metrics with respect to the previous implementation are indicators of this fact. The reason why this happens has been answered in the EDA subsection 4.1.2. The diversity over tags makes this feature extremely noisy, which implies that there are no evident patterns to the algorithm codified on this feature. Moreover, as mentioned in the EDA section 4.1.2, 60% of the tags appear only once in the dataset. Appearing only once on the dataset has several negative implications, the algorithm would not be able to learn the relationship between a particular tag, other features, and the corresponding labeling as it requires a lot of repetition to learn a criterion over a particular feature value. At the same time, is very likely that in the learning stage the classifier has not seen that particular tag and thus could not learn the relationship between that tag and the classification label. This effect has been damped by filtering out tags appearing less than 4 times in the dataset but an appearance frequency of 4 is too small to improve consistency in the data. Other frequency filtering values larger or smaller than 4 have shown worst results.

Metric	Score
Accuracy	0.6934
Recall	0.6812
Precision	0.7308
F1	0.7051

Table 4.7: Version 3 Metrics

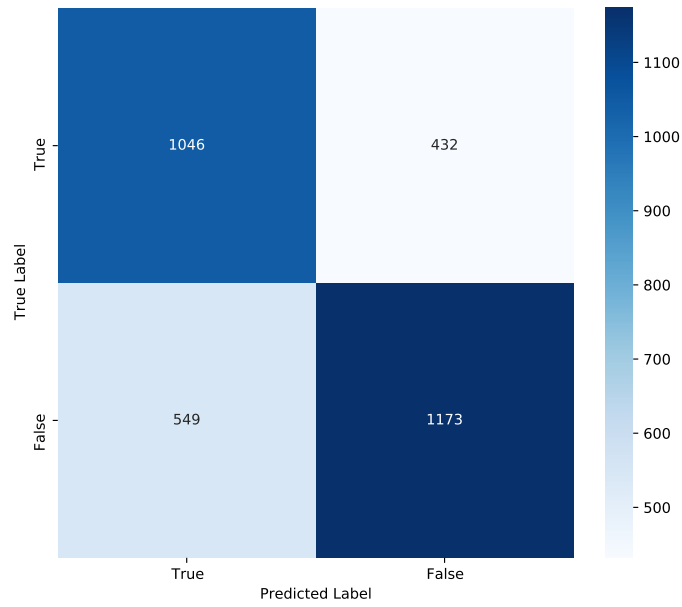


Figure 4.19: Confusion Matrix Version 3

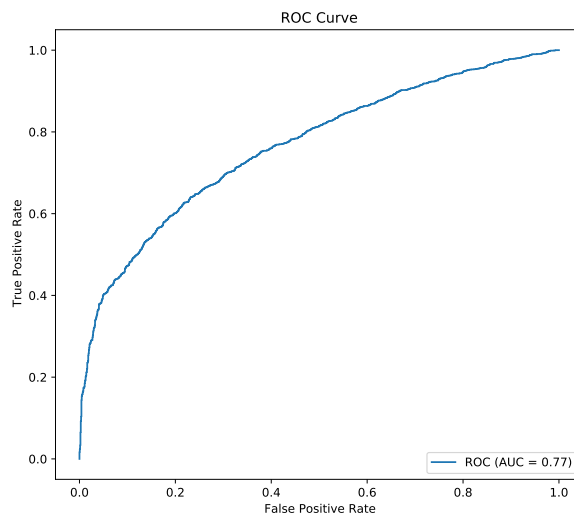


Figure 4.20: ROC Curve Version 3

In order to evaluate the impact of tags in-depth, equal to the previous section, a division of the dataset between most trustworthy authors and most untrustworthy authors has been passed to the model to evaluate the performance. The reason for this evaluation is not only motivated by a mere comparison between the two versions it has further implications. Since those authors appear more frequently it is likely that the topics which the news makes reference to are repeated and thus the tags. Then it is expected that the tags feature plays a role in the decision of this particular news as the algorithm may have been able to learn the underlying relationships. Here are the results.

Top trustworthy authors metrics

Metric	Score
Accuracy	0.7523
Recall	0.0280
Precision	0.6923
F1	0.0537

Table 4.8: Version 3 Trustworthy Authors Metrics

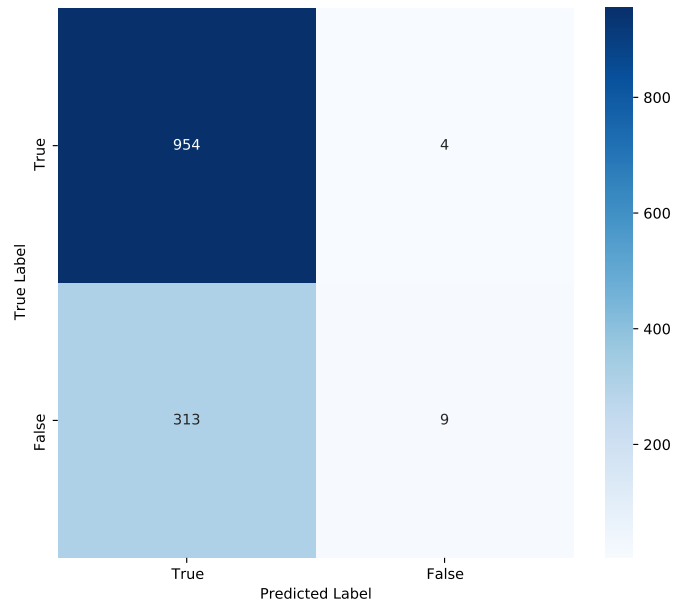


Figure 4.21: Confusion Matrix Version 3 True Authors

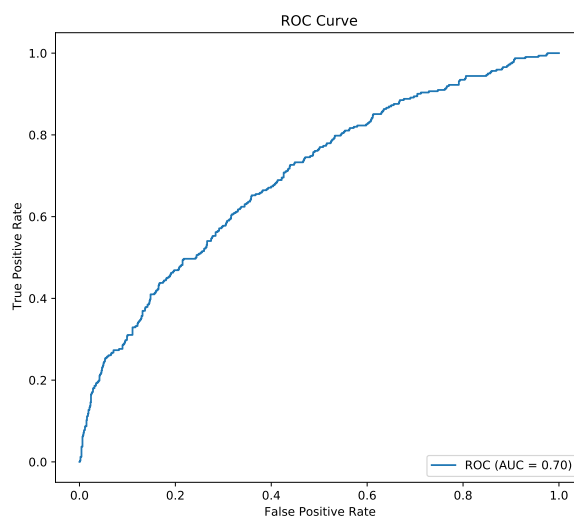


Figure 4.22: ROC Curve Version 3 True Authors

Top untrustworthy authors metrics

Metric	Score
Accuracy	0.8555
Recall	0.9964
Precision	0.8568
F1	0.9214

Table 4.9: Version 3 Untrustworthy Authors Metrics

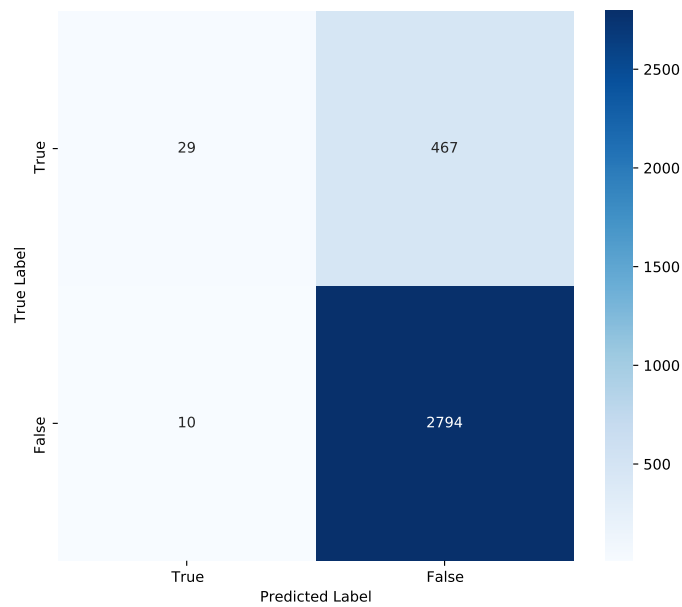


Figure 4.23: Confusion Matrix Version 3 False Authors

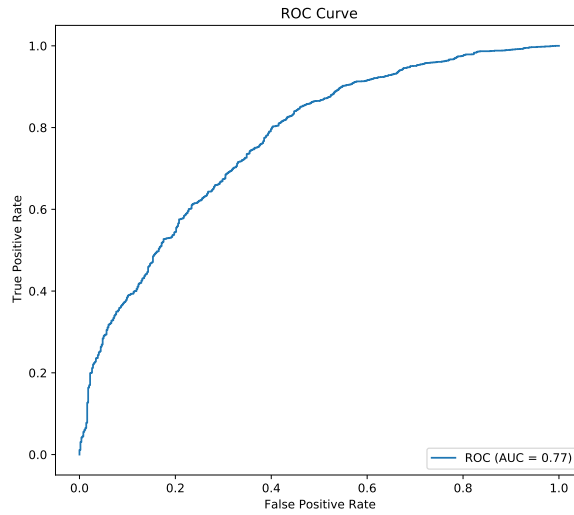


Figure 4.24: ROC Curve Version 3 False Authors

The results show that tags provide almost no useful information. Note that the model has become more rigid since for trustworthy authors the classification tends to label as true more often compared to the previous version. The equivalent happens for untrustworthy authors, the model tends to classify as false more often than in the previous version.

Chapter 5

CONCLUSIONS

The goal of this final bachelor's degree project was to study and develop state-of-the-art techniques based on ML and DL to detect fake news. Two main approaches were implemented.

The first one, and more naive approach, tried to determine the veracity of news using only textual information forming the news article. The implementations in this project which use this approach are the reproduction of FAKEDETECTOR: Effective Fake News Detection with Deep Diffusive Neural Network [38] and the different implementations that I have developed [13] [14] [15].

The following conclusions are drawn from this work. Detecting hoaxes using only news articles textual data is rather cumbersome due to the fact that the model does not have external knowledge to confirm or discuss the veracity of the news. For example, the headline "*Vaccines cause autism*" is a piece of obviously false information to the reader since s/he has external knowledge that proves the headline wrong. The constructed models do not contrast information from other sources thus the headline "*Vaccines cause autism*" or "*Joe Biden is the current president of the US*" have, a priori, the same level of veracity. The implemented models are only able to detect false news with one of the following criteria. The models can assess their criteria based on the stylistic bias of the news. Stylistic bias is the characterizing element that differentiates two texts with the same content but written by two different authors. Then, if fake news has a particular style that characterizes them and is consistent throughout the dataset the model will be able to generate an accurate criterion. In the used dataset the stylistic bias is inexistent since all the textual information provided from Politifact only re-

flects true and false claims written in the same neutral style. Unfortunately, news articles are generally written in a neutral style and are easy to mimic. It can be concluded that stylistic bias is not a strong criterion for fake news detection since is unlikely to appear. The other criteria that a model could use to expose hoaxes in news articles are the detection of keywords, key n-grams, or any combination of words that have a strong correlation with false news. Again this is extremely unlikely to happen since the disparity of news is extensive in the real world. This characteristic of high correlation could only be found in datasets that contain a particularly small subset of all news but clearly, the model using this dataset will fail to generalize.

The study and implementation of the model in the article dEFEND [9] and the study of the paper Mining Disinformation and Fake News [32] revealed a crucial approach for detecting fake news, the power of crowdsourcing. Crowdsourcing consists of using individual opinions to elaborate a general opinion that, in general, captures the socially accepted truth. This solves the problem found in the naive approach of only using the textual information in the news article. People can provide contrasting information that discusses the article and determinates its truthness. On top of that, it also adds explainability to the model. As seen in dEFEND [9] the output of the classifier is not a single probability as it also displays the comments that express conformity or disconformity based on correlation given by an attention layer. Most of the time such comments justify why the news has been classified as true or false.

The ultimate model should use all the elements previously mentioned as well as other indicators such as the broadcasting network of a user in social media, neutral observers like Politifact does and the incorporation of a platform where users can directly report the news to be discussed by models as well as persons.

Fake news is a major threat to the information society in which we are currently living. Truth can not become a malleable concept that can be modified by the rapid broadcasting of disinformation. Fake news is one of the major issues that modern politics is suffering and will suffer as it has become a political weapon. Defamation places as the core element for political success and places morality, responsibility, and accountability in the background.

Bibliography

- [1] Bill Adair. Politifact website. <https://www.politifact.com/>, 2007.
- [2] Braden Hancock Chris Ré Alex Ratner, Paroma Varma and other members of Hazy Lab. The Stanford AI Lab blog weak supervision: A new programming paradigm for machine learning. <https://ai.stanford.edu/blog/weak-supervision/>. Accessed: 09-05-2021.
- [3] APSA. The american political science review. In *The American Political Science Review*. APSA, 1906.
- [4] BBC. A brief history of fake news. <https://www.bbc.co.uk/bitesize/articles/zwcgn9q>, 2017.
- [5] Jason Brownlee. A gentle introduction to the rectified linear unit (relu). <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>, 2019.
- [6] Vanessa Buhrmester, David Münch, and Michael Arens. Analysis of explainers of black box deep neural networks for computer vision: A survey. *arXiv preprint arXiv:1911.12116*, 2019.
- [7] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [8] CIA. Documents from goebbels files (propaganda ministry). <https://www.cia.gov/readingroom/document/cia-rdp81-00706r000100260034-6>, 1947.

- [9] Limeng Cui, Kai Shu, Suhang Wang, Dongwon Lee, and Huan Liu. defend: A system for explainable fake news detection. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2961–2964, 2019.
- [10] Gabriel Graells. defend - code reproduction. <https://github.com/GabrielGraells/Deep-Learning-Techniques-to-address-FakeNews-Detection/blob/main/DEFEND/DEFEND.ipynb>, 2021.
- [11] Gabriel Graells. Exploratory data analysis code. https://github.com/GabrielGraells/Deep-Learning-Techniques-to-address-FakeNews-Detection/blob/main/Exploratory_Data_Analysis/EDA.ipynb, 2021.
- [12] Gabriel Graells. Fakedetector - code reproduction. <https://github.com/GabrielGraells/Deep-Learning-Techniques-to-address-FakeNews-Detection/blob/main/FakeNewsDetector/FakeDetectorModel.ipynb>, 2021.
- [13] Gabriel Graells. Propersed model version 1. https://github.com/GabrielGraells/Deep-Learning-Techniques-to-address-FakeNews-Detection/blob/main/Proposed_Model/Version1/MyModel_V1.ipynb, 2021.
- [14] Gabriel Graells. Propersed model version 2. https://github.com/GabrielGraells/Deep-Learning-Techniques-to-address-FakeNews-Detection/blob/main/Proposed_Model/Version2/MyModel_V2.ipynb, 2021.
- [15] Gabriel Graells. Propersed model version 2. https://github.com/GabrielGraells/Deep-Learning-Techniques-to-address-FakeNews-Detection/blob/main/Proposed_Model/Version3/MyModel_V3.ipynb, 2021.
- [16] Gabriel Graells. Scrapper code. <https://github.com/GabrielGraells/Deep-Learning-Techniques-to-address-FakeNews-Detection/blob/main/Scrapper/scrapper.ipynb>, 2021.

- [17] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.
- [18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [19] Sepp Hochreiter and Jürgen Schmidhuber. Lstm can solve hard long time lag problems. *Advances in neural information processing systems*, pages 473–479, 1997.
- [20] W John Hutchins. The georgetown-ibm experiment demonstrated in january 1954. In *Conference of the Association for Machine Translation in the Americas*, pages 102–114. Springer, 2004.
- [21] Jason S Kessler. Scattertext: a browser-based tool for visualizing how corpora differ. *arXiv preprint arXiv:1703.00565*, 2017.
- [22] Grace W Lindsay. Convolutional neural networks as a model of the visual system: past, present, and future. *Journal of cognitive neuroscience*, pages 1–15, 2020.
- [23] Christopher Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.
- [24] Robin Mansell. The life and times of the information society. *Prometheus*, 28:165–186, 06 2010.
- [25] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [26] Politifact. Politifact news, “as of today, no one has the right to film or photograph mr. biden as he climbs the flight of stairs. new white house spokesman orders!!?”. <https://www.politifact.com/factchecks/2021/apr/02/tiktok-posts/no-media-isnt-banned-filming-biden-he-climbs-stair/>, 2021.
- [27] Politifact. Politifact news, “the global cost of corruption is as much as 5% of the world’s gdp.”. <https://www.>

- politifact.com/factchecks/2021/may/07/kamala-harris/kamala-harris-said-corruption-costs-much-5-worlds-/, 2021.
- [28] Yenumula B Reddy, Nandigam Gajendar, Portia Taylor, and Damian Madden. Computationally efficient resource allocation in ofdm systems: Genetic algorithm approach. In *Fourth International Conference on Information Technology (ITNG'07)*, pages 36–41. IEEE, 2007.
 - [29] Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.
 - [30] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.
 - [31] Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, and Huan Liu. Fakenewsnet: A data repository with news content, social context and spatialtemporal information for studying fake news on social media. *arXiv preprint arXiv:1809.01286*, 2018.
 - [32] Kai Shu, Suhang Wang, Dongwon Lee, and Huan Liu. Mining disinformation and fake news: concepts, methods, and recent advancements. In *Disinformation, Misinformation, and Fake News in Social Media*, pages 1–19. Springer, 2020.
 - [33] Kai Shu, Suhang Wang, and Huan Liu. Exploiting tri-relationship for fake news detection. *arXiv preprint arXiv:1712.07709*, 8, 2017.
 - [34] Kai (Arizona State University) Shu. Fakenewsnet - harvard dataverse. <https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/UEMMHS>, 2019-01-14.
 - [35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
 - [36] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending against neural

- fake news. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 9054–9065. Curran Associates, Inc., 2019.
- [37] Jiawei Zhang. Get rid of suspended animation problem: Deep diffusive neural network on graph semi-supervised classification. *arXiv preprint arXiv:2001.07922*, 2020.
- [38] Jiawei Zhang, Bowen Dong, and S Yu Philip. Fakedetector: Effective fake news detection with deep diffusive neural network. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1826–1829. IEEE, 2020.