
## UE905 EC2 Software Project

# OAKS
## Online App for Knowledge Substantiation

Submitted in partial fulfillment
of the requirements for the degree of
**MSc Natural Language Processing**

Anna Mosolova, Elisa Lubrini

February 2022

UNIVERSITÉ DE LORRAINE

Institut des sciences du Digital
Management & Cognition
COMPOSANTE DE L'UNIVERSITÉ DE LORRAINE

# Abstract

Given the influence of news pieces on the public's opinions and beliefs, the lack in accuracy and truthfulness of certain information shared by media outlets continues to be a major ethical concern.

Although a number of computational approaches have been proposed to face these issues by detecting "fakeness" in news pieces, the state of the art is, at the time of writing, still far from achieving accurate detection and the research for a better solution is still ongoing.

Our proposed solution takes advantage of existing fake news detection approaches and combines them into an ensemble to produce a single prediction. The four approaches we explored are: feature-based classification, neural networks, cross-checking, and knowledge bases.

The current output consists in a single prediction score. For the testing we used the pipeline as a binary classifier; while the end users sees the classification in the form of a percentage.

# Contents

# Chapter 1

# Introduction

"Fake news" is a neologism used to describe news pieces, in their various forms (articles, broadcasts, etc.), that contain false or misleading information. According to [26], seven main types of fake news have been identified:

1. *satire or parody*: no intention to cause harm but has potential to fool;
2. *false connection*: when headlines, visuals or captions do not support the content;
3. *misleading content*: misleading use of information to frame an issue or an individual;
4. *false context*: genuine content is shared with false contextual information;
5. *impostor content*: genuine sources are impersonated with false, made-up sources;
6. *manipulated content*: genuine information or imagery is manipulated to deceive, as with a "doctored" photo;
7. *fabricated content*: new content is 100% false, designed to deceive and do harm.

Additionally, [18] mentions the importance of non-linguistic content, such as visual content (photos, videos, etc.), as a potential contributor to the spread of false information[1].

Given the influence of news pieces on the public's opinions and beliefs, the lack in accuracy and truthfulness of certain information shared by media outlets continues to be a major ethical concern.

[1] In this project, we will only target issues related to purely linguistic content, as non-linguistic features fall outside the scope of the framework in which this project was developed. However, systems processing other types of media could constitute an interesting feature to be added to our application, in the future (see Section 5.1).

Although a number of computational approaches have been proposed to face these issues by detecting "fakeness" in news pieces, the state of the art is, at the time of writing, still far from achieving accurate detection, and the research for a better solution is still ongoing. For more details on the already developed methods, refer to Chapter 2.

Our proposed system takes advantage of existing fake news detection approaches and combines them into an ensemble to produce a single prediction. The four approaches we explored are: feature-based classification, neural networks, cross-checking, and knowledge bases.

# Chapter 2

# Related Work

A variety of theories and implementations tackling fake news detection have been proposed in the literature so far. In the following sections, we cover some approaches to fake news detection that constitute the basis to our own work, i.e., feature-based rules, cross-checking, knowledge base, and neural network approaches.

## 2.1 Fake News Features

A number of linguistic features have been identified as potential telling signs for a piece of news of being fake. The following classes of features have been analysed: *Syntax-based*, *Sentiment-based*, *Grammatical-based*, and *Readability-based* [3].

These features were used to train an Ensemble of (1) Gaussian Naive Bayes, (2) Kernel Naive Bayes, (3) Linear SVM, and (4) Gaussian SVM obtaining an overall accuracy of 72%.

## 2.2 Knowledge-based Fact Checking

Computationally, knowledge is often encoded in sets of semantic triples, following a subject-predicate-object structure standardised by the Resource Description Framework (RDF) [13].

### 2.2.1 An Ideal System

The automatisation of knowledge-based fact checking relies on information retrieval, graph theory, and NLP [4]. An ideal pipeline would consist in two steps: fact extraction and fact checking, defined as follows [27].

**Fact Extraction** This step consists in extracting knowledge, often from raw texts, and pre-processing it into a consistent graph.

**Fact Checking** Once a knowledge representation of the content of an article is obtained, the next step is to compare it with a reference knowledge base that is known to hold accurate information. This comparison should result in an "authenticity index" in the form of a percentage[1].

### 2.2.2 Related Implementations

Most existing works on fake news detection are based on the analysis of the writing style of news content as features to classify news articles [8, 7, 25, 11]. In spite of the high scores achieved during evaluation, these methods cannot explain what makes each news piece fake or not. Conversely, fact checking, also known as content-based fake news detection, appears to be more "aware" of how the content influences the prediction [22].

A possible implementation of a fact checking system [17] consists in building two training sets composed of a knowledge graphs each, one containing trustworthy news and the other fake news. Each of these sets can then be used to train a TransE [2] model, and a news item is verified by comparing the bias of the two models over the extracted relations.

More recently, DEAP-FAKED [16] proposed a BiLSTM trained on the Kaggle Fake News[2] dataset to create article representations. Classification with a multilayer perceptron achieved a F1 score of 88.66%. However, this system achieved its highest performance when only taking into account news title, which, again, is a sign of its poor comprehension of the articles' overall content.

Lastly, [10] proposed transforming fake news detection into a knowledge graph classification problem, leveraging developments on graph neural networks (GNNs) and their "superior performance on non-Euclidean data".

## 2.3 Neural Networks

Various approaches to fake news detection using neural networks have been proposed in the past years. Among the most recent approaches, [15] attempted to use Text-transformers for the

[1] For these steps in more details, including formal definitions and formulas for computation, refer to [27] *Section 22*.

[2] `https://www.kaggle.com/c/fake-news/overview`

purpose of COVID-19 fake news detection. The authors presented a solution for the shared task *COVID19 Fake News Detection in English*[3]. Their submitted system consisted of an ensemble of several models, such as BERT, Ernie, RoBERTa, XL-Net, and Electra, using pseudo-labeling technique as training method. The model achieved an F1 score of 98.59%.

[3] `https://competitions.codalab.org/competitions/26655`

# Chapter 3

# Methodology

The system assembled during our project consists of an ensemble of four different modules, corresponding to four different approaches (feature-based classification, neural networks, cross-checking, and knowledge bases) (see Figure 3.1). The ensemble leverages the results of each approach to give more accurate results than those that the four separate modules taken individually would give.

The application[1] was developed in Python 3[2]. The main libraries used in the project are: PyTorch[3], scikit-learn[4], pandas[5], spaCy[6] and Flask[7].

In the following sections, the pipeline (Section 3.1) and the dataset (Section 3.2) will be described, as well as the four modules integrated in our proposed system (Sections 3.3.1, 3.3.2, 3.3.3, 3.3.4). Section 3.5 provides information on the front-end part.

[1] Source code and documentation are fully available on GitHub, at `https://github.com/e-lubrini/oaks`.
[2] `https://www.python.org/`
[3] `https://pytorch.org/`
[4] `https://scikit-learn.org/stable/`
[5] `https://pandas.pydata.org/`
[6] `https://spacy.io/`
[7] `https://flask.palletsprojects.com/en/2.0.x/`

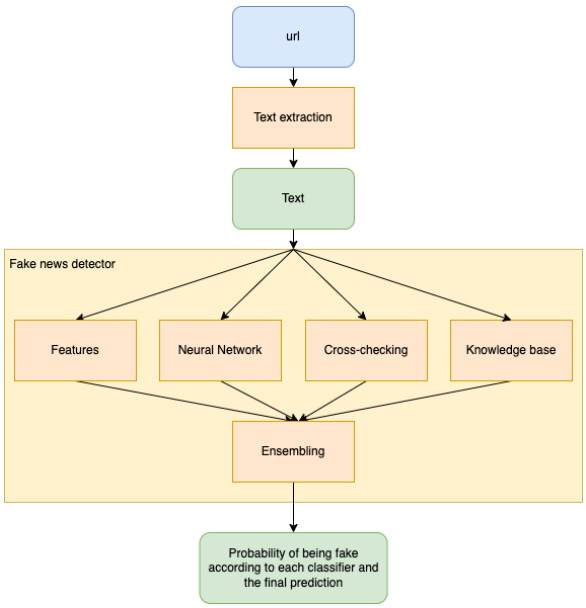


Figure 3.1: Fake news detector modules

## 3.1 Pipeline

**Input** The application we developed takes an URL as an input and extracts the text of the article found at such URL.

**Algorithm** The algorithm consists in an ensemble (see Section 3.4) of four different modules, described in the following sections[8].

**Output** The output of the app is a percentage of the likelihood that the article found at the given URL is fake. Future improvements could include providing more detailed explanations on how each statement in the article contributed to the outputted likelihood (see Section 5.1 for more information).

## 3.2 Data

While exploring the datasets dedicated to fake news detection, we found that most of the datasets suggest solving the task of stance detection[9], instead of fake news detection. For our

[8] At the time of writing, the knowledge-base module is not yet integrated in the pipeline, temporarily leaving only the three remaining modules.

[9] For example, the dataset prepared for the RumourEval task at SemEval 2017: `https://github.com/autoreleasefool/rumoureval`.

purposes, we had firstly chosen the LIAR dataset, but then decided to opt for LIAR-PLUS[10] as the former one does not have publicly available texts.

As for the LIAR-PLUS dataset, the overall statistics about it is presented in Table 3.2.

| | |
|---|---|
| Training set | 10,269 sentences |
| Validation set | 1,284 sentences |
| Test set | 1,283 sentences |
| Main topics | politics, economy, healthcare, taxes, federal, budget, education, jobs, state budget, candidates biography, elections, and immigration |
| Average sentence length | 20 tokens |
| Number of **false** news | 1995 sentences |
| Number of **real** news | 1676 sentences |

Table 3.1: Statistics concerning the LIAR-PLUS dataset

From the table, we can notice that the average sentence length is pretty small. This makes the average unit resemble a titles more than an actual article. Although this does not affect cross-checking and feature-based modules, a solution needed to be found to train the neural network module. The two options were either breaking the text into smaller pieces and average the results (the strategy chosen for time being) or train another model with the bigger maximum length. A solution that is yet to be integrated is that of finding and using a corpus that resembles more the body of an article, rather than its title. For more information on how we are planning to change the dataset, see Section 5.1.

## 3.3 Modules

This section presents the four modules implemented so far.

### 3.3.1 Feature-based Approach

#### Features

As fake news have become more and more widespread in various news sources, they attracted the attention of the authorities. In general, fake news are able to produce turmoils and conflicts, so all governmental structures are interested in decreasing the number of fake news. For these purposes, forensic linguists have been developing various methodologies to detect the presence of fake news in the news sources. One of such studies [19] analyses real and fake news articles and lists the following features that could help in the identification of the fake texts.

[10] `https://github.com/Tariq60/LIAR-PLUS`

**Punctuation errors** *Reasons:* (1) the author might be poorly educated; (2) an author is pretending to belong to a certain social group; (3) the news is a translation from a language with different grammatical rules.

*Example: A person that I saw yesterday, was here today.*

**Lexical repetitions** *Reasons:* (1) amplification of the emotional impression from a text; (2) highlighting a particular part of the information for a reader.

*Example: Finally, the disaster came to our region as well. To our places, where there is still some fresh air, fresh soil, fresh water, but it's not for long....*

**Plural forms** *Reasons:* (1) concealing the source of the information; (2) imitation of the view of the majority.

*Example: People are talking about this event a lot.*

**Loaded language** *Reasons:* (1) making people want to share the emotions they felt when they were reading a text.

*Example: I cannot talk to everyone about what's going on, so I'm writing here. Or even screaming. I'm trying my best to save the beloved ones.*

**Misrepresentation of fact** *Reasons:* (1) making a text sensational.

*Example: Ministry of health listed the vaccines that could lead to death in a month.*

**Links to an unreliable source** *Possible sources*: (1) some person from the authorities; (2) a relative of the speaker; (3) a relative of the speaker working in the governmental organization. All these three sources appeared when people lose faith in the official information, so they tend to believe to such sources.

*Example: Guys, I just talked to the mother of my husband. She is working in the hospital and she said there were going to introduce the lockdown from the next week!*

**Rhetorical questions** *Reasons:* (1) Intensifying personal involvement of the reader in the discourse; (2) drawing the reader's attention to the subject; (3) amplification of emotional impression from a text.

*Example: Do you want to be a failure for the rest of your life?*

**Excessive usage of exclamation marks and imperative mood** *Reasons:* (1) amplification of emotional impression from a text.

*Example: Drink more hot water and stay home!*

**Spelling mistakes** *Reasons:* (1) the author might be poorly educated; (2) the author is pretending to belong to a certain social group
*Example: Coronovirus is more dangerous for cats than for humans!!!*

**Contrast of *we* vs. *they* stances**

*Reasons:* (1) creation of a bad impression of one of the sides of the conflict.

*Example: They are monsters who are stealing our treasures and we cannot even complain about it!*

**References to past events** *Reasons:* (1) manipulating the feelings of people about the past events, so that they can reflect them on the current ones.

### Implementation

We implemented most of the aforementioned features, leaving out rhetorical questions, links to the unreliable sources, and misrepresentation of the facts, for different reasons. The existing solutions for rhetorical questions are not performing well enough, while the misrepresentation of facts seems to be hardly possible to evaluate using any modern NLP models. Lastly, checking the links to the unreliable sources would require a deep semantic analysis that is not accessible using the current semantic analysis approaches.

For the implementation of the remaining modules we used several tools for automatic natural processing:

- Spelling mistakes: *textblob*[11] to check the spelling of each word in a text;
- Lexical repetitions: counting how many words were repeated in the text;
- Plural forms: retrieving the part of speech of the subject of a sentence and checking if it is a part of the following list: *we, they, us, them*;
- Loaded language: *spacytextblob*[12] to check the polarity of the text;

[11] `https://textblob.readthedocs.io/en/dev/`
[12] `https://spacy.io/universe/project/spacy-textblob/`

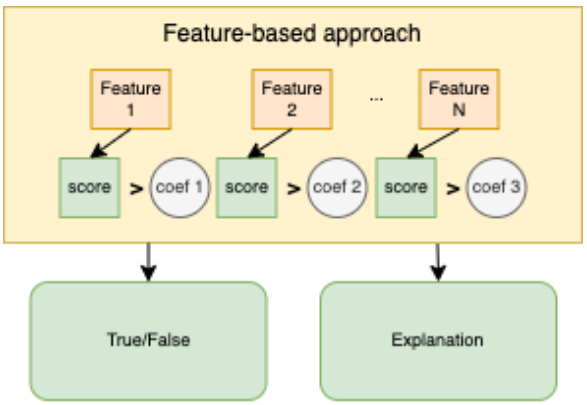


Figure 3.2: Feature-based module

- Excessive usage of exclamation marks and imperative mood: counting how many exclamation marks and their combinations are present in a text;
- Punctuation errors: *fastpunct*[13] package to check the punctuation errors in a text;
- References to past events: checking how many verbs are in the past tense.

The whole feature-based module is shown in the Figure 3.2.

#### Thresholds

The seven implemented features produce different types of output: *references to past events* and *loaded language* output a float, while others output an integer. In order to improve the results of this module, we tuned the threshold for each of these features. This way, if the output value is below a certain threshold, the news is considered fake. For these purposes, we checked the results for the values from -0.3 to 0.5 for the *loaded language* feature, from 0.0 to 1.0 with the step of 2 for the *references to past events*, and all values from 0 to 20 with the step of 2 for the remaining features. The computations were done on the training part of the dataset (see Figure 3.3). Then, among the best combinations, we chose the one that produced the best results on the validation dataset.

### 3.3.2 Cross-reference Approach

One of the ways to check if some news is fake is to find other sources that convey the same idea about the event. In order to do this, we created another module dedicated to cross-reference checking. For these purposes, we use the GoogleNews python library by means of

[13] `https://github.com/notAI-tech/fastPunct`

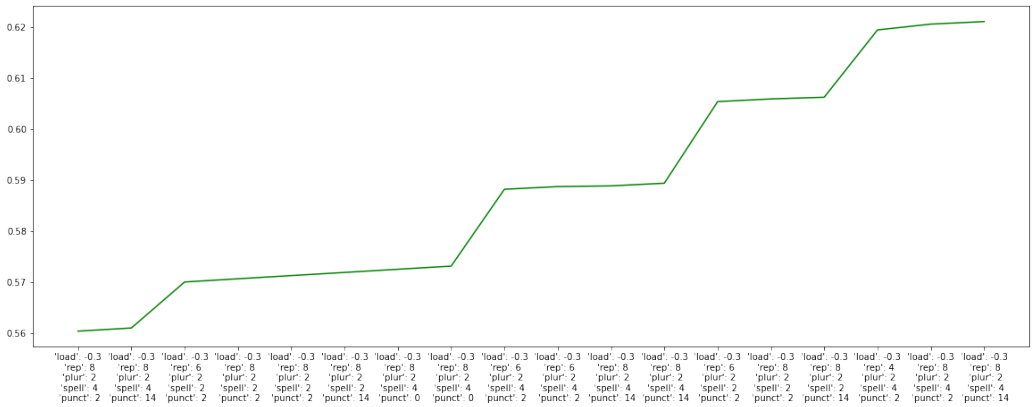


Figure 3.3: Hyper-parameters search for Feature-based Module

which we extract the titles of all articles that match our searching query (the original text). After extracting these titles, we convert both them and the original text into embeddings and compare them using cosine similarity. If the average cosine similarity between the titles and the original text is higher than a certain threshold, we state that this text is real, and that it is false otherwise.

In the description of the model, it is possible to notice several possible issues. Firstly, the issue with the long texts that could be used as an input to this module. To tackle this problem, we added the summarisation model that summarise every text that is longer than a certain threshold. Concerning the summarisation model, we tried several popular neural network based summarisation approaches such as several variations of Pegasus and BART models, and BigBirdPegasus. In addition to this, we also tried the graph-based TextRank approach. After manually checking the summarisation results, we came to a conclusion that NN-based models do not summarise correctly the texts of the politics domain (which was the main domain of our dataset), so they cannot be used for the summarisation purposes on this dataset. Among all these models, Pegasus xsum produced the results that were better than the other, so we left it as a possible model to use in this module, however, the defualt one for now is TextRank.

Another possible issue is the choice of a language model to use to produce the sentence embeddings. We tried several SOtA models such as BERT, BERT-large, RoBERTa, Albert, Albert-large and compared them on the basis of cosine similarity scores they produce. The results are shown in the Figure 3.4. After analysing the results we came to a conclusion that all models tend to overestimate the closeness of two texts, so we decided to take the model that produces the lowest cosine similarity which is an Albert-large model.

The overall pipeline for the cross-reference module is shown in the Figure 3.5.

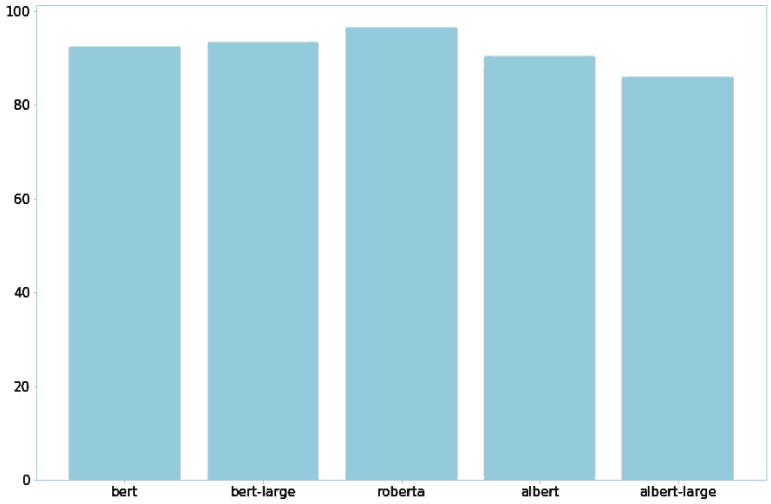


Figure 3.4: Language models comparison based on the produced similarity scores

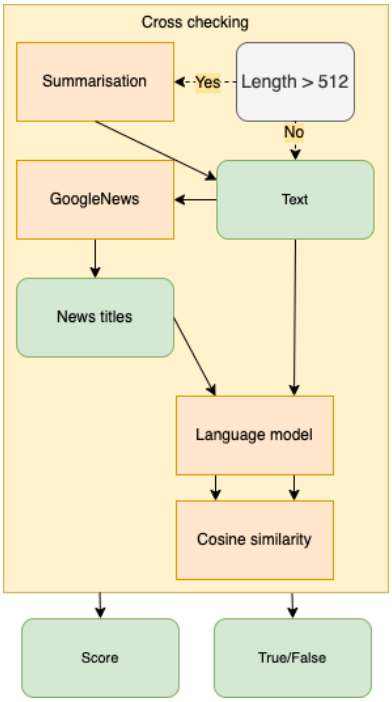


Figure 3.5: Cross-checking approach

### 3.3.3 Neural Network Approach

Nowadays, the majority of approaches used to solve various NLP tasks is exploiting the neural networks. Previous attempts to build a system that could detect fake news also included various nerual network based approaches.

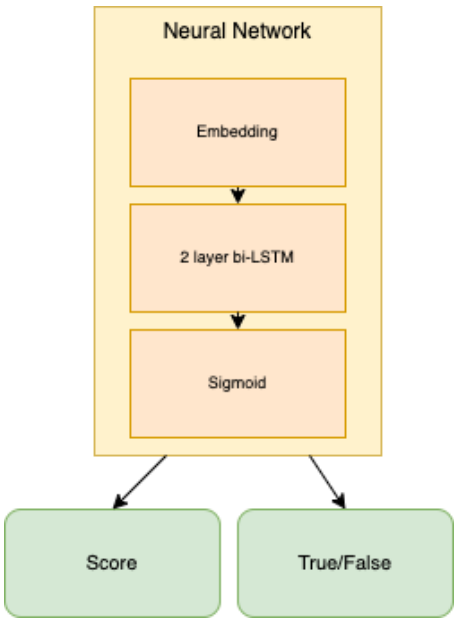


Figure 3.6: Neural Network approach

We decided to implement several popular models such as convolutional neural network, recurrent neural networks with GRU, LSTM and RNN layers as well as BERT and roBERTa based classifiers.

In addition to this, we also tested the solution applied to solve the COVID-related fake news detection task[14] This solution consisted in using pseudo-labeling to fine-tune several Transformers models such as two versions of COVID-BERT, DistilRoBERTa, ERNIE and COVID-SciBERT. For our experiments we only mention the results obtained using DistilroBERTa and ERNIE.

The best results obtained after the training and tuning the parameters of all models are shown in the table 3.3.3.

| Model | F1-weighted |
|---|---|
| LSTM | 68% |
| GRU | 60% |
| CNN | 60% |
| GRU with BERT embeddings | 53% |
| BERT | 60% |
| roBERTa | 61% |
| DistilRoBERTa fake news detector[15] | 65% |
| ERNIE fake news detector[15] | 64% |

Table 3.2: Observed results when implementing different architectures

The final architecture is presented on the Figure 3.6.

[14]https://github.com/archersama/3rd-solution-COVID19-Fake-News-Detection-in-English

### 3.3.4 Knowledge-base Approach

The aim of this module is to improve the score of the ensemble by leveraging verified data encoded in knowledge bases. This allows us to check the veridicity of data contained in an article when encoding it in a standard format, i.e a set of triples.

#### Background

An ideal system would extract triples from the inputted article and compare them to a reference KB to check for inconsistencies [27].

After some research we came to the conclusion that this solution would have been impossible to implement due to the scarcity of data in even the largest knowledge bases (e.g. DBpedia [14] or Wikidata [24]). Such limitations are especially important in the news field, as news pieces can often refer to specific factual information, which veridicity might depend on details not mentioned in the reference KB [17].

This led us to ask some academics for advice on how to proceed. One of our most prolific discussions was with the co-author of a paper on fake news detection (Mathieu d'Aquin, [5]), who suggested implementing the following algorithm.

#### Algorithm

Given a set of $n$ triples present in the reference database $D = \{\{ds_i, dp_i, do_i\} \mid i \leq n\}$ and a set of $m$ triples extracted from the inputted article $A = \{\{as_j, ap_j, ao_j\} \mid j \leq m\}$, we want to derive:

- for each triple in the article, all triples in the reference database that share the subject and the object with the former; namely, all $E_i \subseteq D$ such that

$$E_i = \{\{ds_i, dp_i, do_i\} \in D \mid ds_i = as_j, do_i = ao_j, i \leq n, j \leq m\}$$

- for each triple in the article, all triples in the reference database that share the class of the subject and the class of the object with the former; namely, all $C_i \subseteq D$ such that

$$C_i = \{\{ds_i, dp_i, do_i\} \in D \mid p(ds_i) = p(as_j), p(do_i) = p(ao_j), i \leq n, j \leq m\}$$

In order to compute the likelihood for triples in an article to be truthful, we will compute the ratio between: (1) the number of triples in the reference database containing the mentioned

objects and subjects, and (2) the number of triples in the reference database the classes of whose elements corresponds to that of the objects and subjects in the extracted triple, with each ratio combined through multiplication as follows.

$$\prod_{j=1}^{m} \frac{|E_i|}{|C_i|}$$

**Tools and Resources**

The first step of the pipeline is to divide the text into sections to be pre-processed by a text-to-rdf converter. The tool chosen for this step is FRED[15] [6], accessed through its Python API.

The second step is to derive the classes from the tokens based on a reference dataset. In this case, we used the DBpedia database [23], extracting classes using the `SPARQLWrapper` [20] package to access Sparql from Python.

## 3.4 Ensembling

Following the idea of "wisdom of the crowd"[21] that underlies the ensemble learning strategy, we are using the modules ensembling in this work in attempt to improve the overall score of all models. For these purposes, we tested two types of ensembling:

- The majority voting which consists in predicting the class for which the majority of models "voted" with the True value;
- Logistic Regression ensembling which consists in using the probabilities produced by the modules as features for a Logistic Regression model that is trained to produce the final result.

The whole pipeline is provided in the Figure 3.7.

[15] http://wit.istc.cnr.it/stlab-tools/fred/

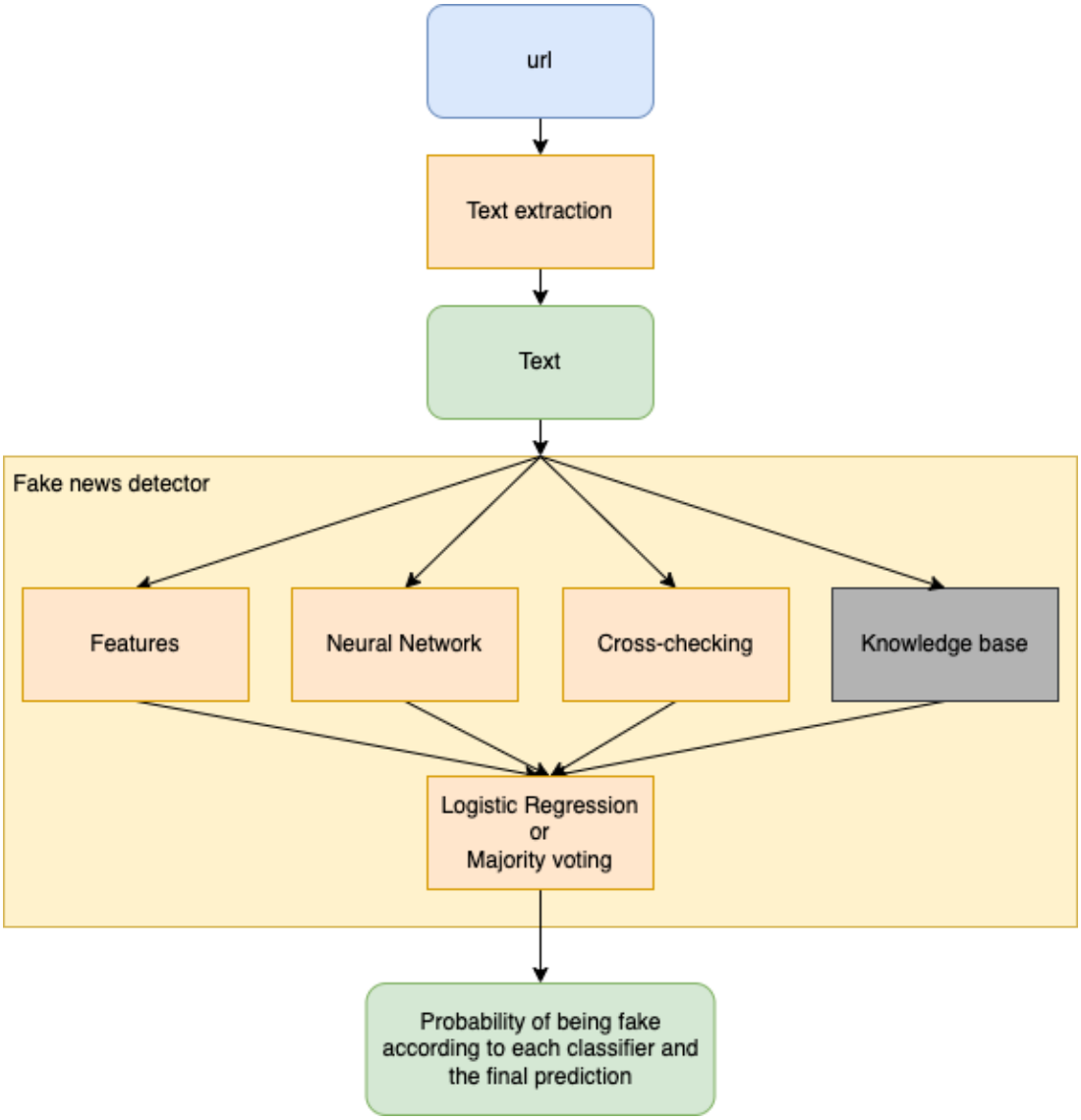


Figure 3.7: Ensemble architecture

## 3.5 Graphic User Interface

For the graphic user interface, we opted to deploy a web app, in order to avoid possible cross-platform compatibility issues, which resolution is outside of the main scope of this project.

### 3.5.1 Structure

The app is currently structured as two main webpages: one requiring the user's input and briefly describing the features and specifications of the app (Figure 3.9); the other displaying the results outputted by the pipeline (Figure 3.8).

### 3.5.2 Tools

The web app was developed using the Flask framework[16] [9] for web app development in Python. The web app prompts the user to enter a url, which is then processed by the algorithm and returns the obtained percentage in a new page. Future improvements include the addition of more information to be displayed for algorithmic transparency and Better understanding of the results (see Section 5.1).


Figure 3.8: Page displaying the results

[16] https://flask.palletsprojects.com/en/2.0.x/

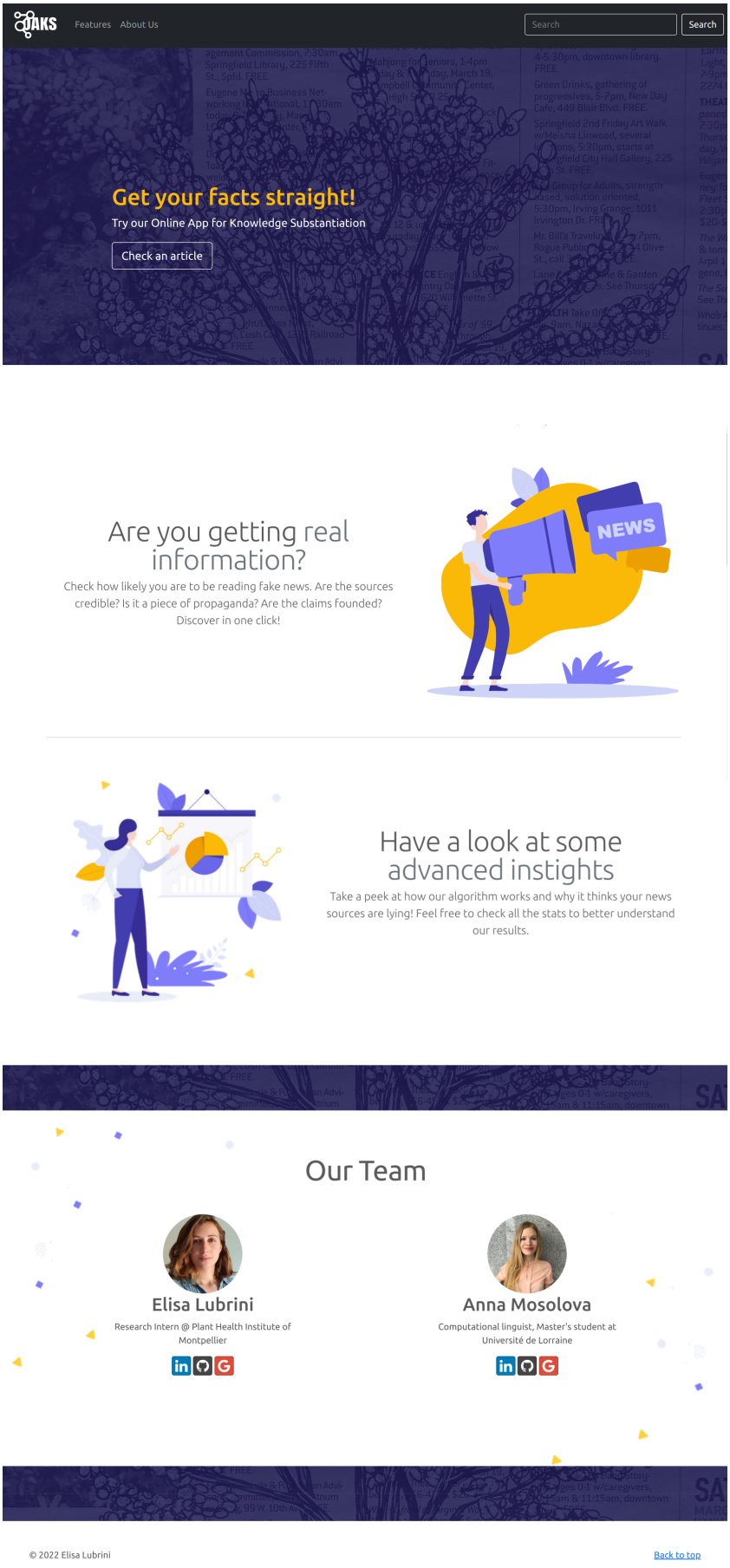


Figure 3.9: Web app homepage

# Chapter 4

# Results and Discussion

## 4.1 Evaluation

Each module produces different results to be used in the ensemble.

Feature-based module produces the binary value (True/False) and the explanation of the decision from each feature in the module. Neural network module returns the binary value and the probability of some news to be fake. Cross-reference module produces the binary class and it probability. It is also able to return the titles of the most similar news found on GoogleNews. For the following evaluation we used either the probabilities or the binary values returned by the systems.

For the evaluation of the whole system and each part of it we used a commonly used metric - F-score. In the table 4.1, the results for each module and for two types of ensembling are shown, the corresponding confusions matrices are provided in the Figure 4.1.

From the results we can see that neural network module works better than all other modules and show the same accuracy as the voting ensembling.

| Metric | Voting | Logistic Regression | Feature-based | Cross-reference | Neural network |
|---|---|---|---|---|---|
| F1 for fake news | 0.65 | **0.68** | 0.61 | 0.60 | 0.62 |
| F1 for real news | 0.42 | 0.31 | 0.35 | 0.43 | **0.53** |
| F-micro | **0.58** | 0.57 | 0.51 | 0.53 | **0.58** |
| F-macro | 0.54 | 0.50 | 0.48 | 0.52 | **0.57** |
| F-weighted | 0.55 | 0.51 | 0.49 | 0.53 | **0.58** |

Table 4.1: The quality of the modules and the whole system

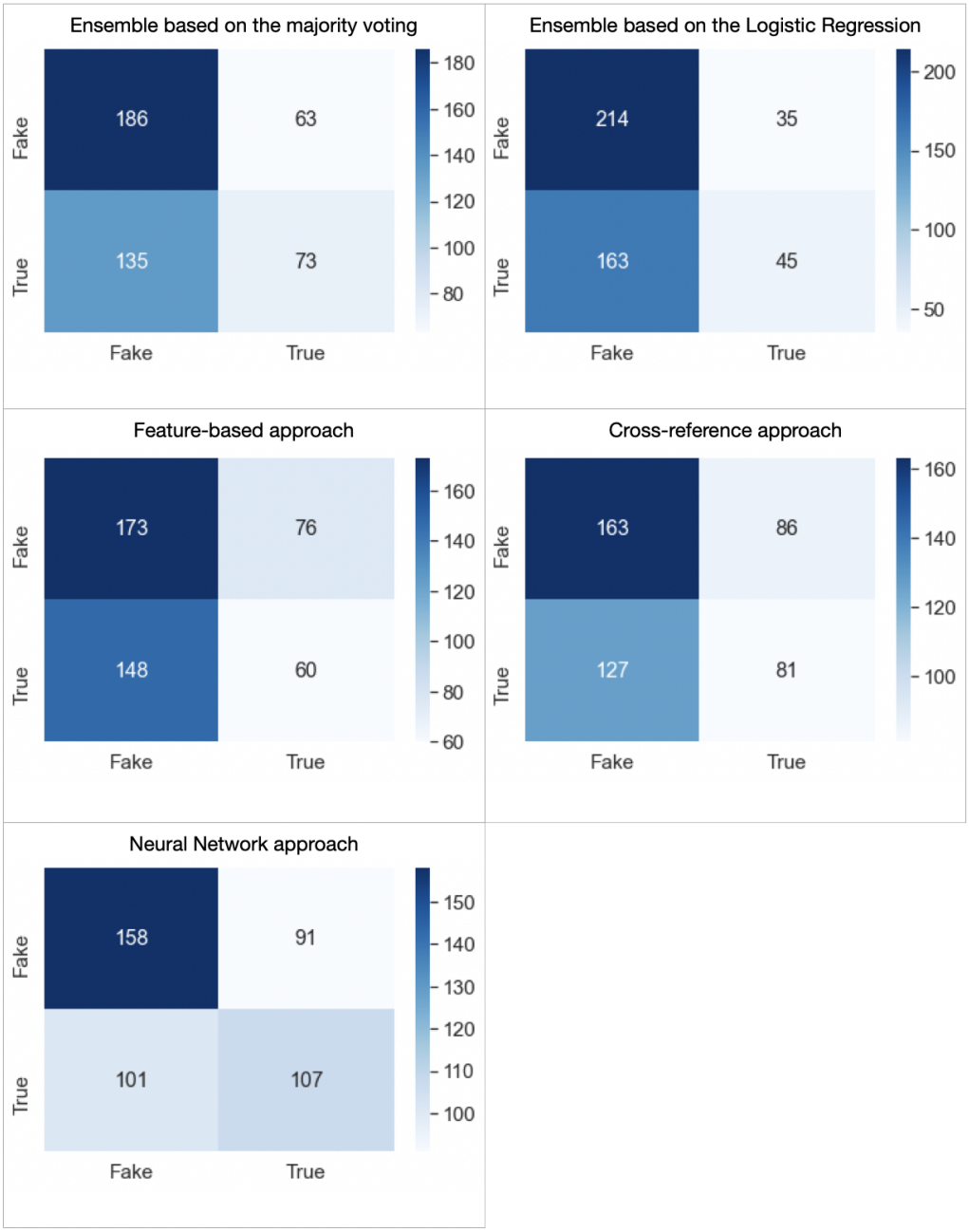


Figure 4.1: Results for 3 modules and different types of ensembling

## 4.2 Discussion

The ensemble of the results presented in the previous sections show that fake news detection task is difficult to solve even using various points of view (pragmatic, linguistic and statistical ones).

The linguistic side of the problem showed that sometimes it is hardly possible to detect some linguistic features of the texts, especially the ones related to the deep level of semantics of the

text. However, it was also shown that references to past events and lexical repetitions tend to appear more frequently in the fake texts rather than in real ones. The reason for this could consist in the fact that it is easier to manipulate people when they are vulnerable because of feeling the emotions coming from the past events or the emotions that were imposed on them by repeating them many times.

The use of statistics implemented in the form of a neural network has shown a strong performance on this task. However, statistical methods of fake news detection do not take into consideration the knowledge of the world when taking a decision, so they cannot be considered as fully reliable for such a sensible task, despite their strong performance.

Tackling this problem from the pragmatic point of view has also revealed several precious insights about the task. Among them, we could point out that language model tend to put sentences to closely to each other, so it could be difficult sometimes to understand whether their similarity is real or it is caused by some factors not related to the sentences' semantics.

# Chapter 5

# Conclusion

In our project, we explored different approaches to fake news detection and ensembled them into a single model for binary classification. The four modules contributing to the ensemble represent each a distinct approach to fake news detection, i.e. feature-based classification, neural networks, cross-checking, and knowledge bases.

The resulting product is a web app for fake news detection in the process of being published.

## 5.1 Future Work

During the upcoming months we will be working on improving the functionality of the app according to the following points.

### 5.1.1 Dataset

The main downside of the currently used dataset is the extremely short size of each of the training corpora, in contrast with the size of normally processed articles. In order to improve the training, a better dataset for fake news detection composed of full articles has been found, and will soon integrated in the pipeline, substituting the one currently used for training and testing.

### 5.1.2 Result Information

The main milestone to be achieved in the upcoming months is the implementation of a more detailed result report that allows the user to have more insight into which modules gave which

results and which statements in the articles contributed the most to the fakeness index.

### 5.1.3 Knowledge-base Module

A major part of of the upcoming work will be dedicated to finish the knowledge base module and integrate it into the pipeline. Additionally to the algorithm in the process of being implemented, other more time-consuming approaches that could not fit in the current project timeframe will also be considered.

### 5.1.4 Graphic User Interface

**Loading Page** Given the amount of time taken by the pipeline to process the input and return even the most basic insights, a web element or page that signals to the user that the output is in the process of being computed would largely improve the user experience.

**Model specific insights** Given the poor results of the overall classification tool, an output that is more interpretable than the mere classification/percentage would become the main and most valuable feature of the web app. The main idea behind the improvement of the displayed results is showing the text of the article with a coloured highlighting on the sentences that have been detected as contributing to the article's "fakeness." This would turn the app from a completely automatic pipeline to be blindly trusted into an aiding tool for the newsreader to improve their critical reading.

**Interactive model filtering** To make the pipeline even more transparent and make the user take part in the computation, we could integrate an interactive interface allowing the user to selectively activate the modules of the ensemble and compare the difference in how each of them, with its strength and weaknesses, analyses the inputted document.

### 5.1.5 Additional Approaches

Additional approaches have been recently explored and could be integrated in our project in the future. We found particularly interesting those incorporating: Discourse Analysis [1], Propagation-based Analysis [5], and User-based Analysis [12] and are planning to review their methods shortly.

# Bibliography


[1] L. Azevedo, M. d'Aquin, B. Davis, and M. Zarrouk. Lux (linguistic aspects under examination): Discourse analysis for automatic fake news classification. In *ACL/IJCNLP (Findings)*, pages 41–56, 2021.

[2] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[3] A. Choudhary and A. Arora. Linguistic feature based learning model for fake news detection and classification. *Expert Systems with Applications*, 169:114171, 2021.

[4] S. Cohen, J. Hamilton, and F. Turner. Computational journalism. *Commun. ACM*, 54:66–71, 10 2011.

[5] De keersmaecker, Jonas and Roets, Arne. 'Fake news' : incorrect, but hard to correct : the role of cognitive ability on the impact of false information on social impressions. *Intelligence*, 65:107–110, 2017.

[6] A. Gangemi, V. Presutti, D. R. Recupero, A. G. Nuzzolese, F. Draicchio, and M. MongiovÃ¬. Semantic Web Machine Reading with FRED. *Semantic Web*, 8(6):873–893, 2017.

[7] S. Gilda. Notice of violation of ieee publication principles: Evaluating machine learning algorithms for fake news detection. In *2017 IEEE 15th Student Conference on Research and Development (SCOReD)*, pages 110–115, 2017.

[8] M. Granik and V. Mesyura. Fake news detection using naive bayes classifier. In *2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)*, pages 900–903, 2017.

[9] M. Grinberg. *Flask web development: developing web applications with python.* " O'Reilly Media, Inc.", 2018.

[10] Y. Han, A. Silva, L. Luo, S. Karunasekera, and C. Leckie. Knowledge enhanced multimodal fake news detection, 2021.

[11] Y. Jiang, Y. Liu, and Y. Yang. Languagetool based university rumor detection on sina weibo. In *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 453–454, 2017.

[12] Z. Jiawei, L. Cui, Y. Fu, and F. Gouza. Fake news detection with deep diffusive network model, 05 2018.

[13] O. Lassila and R. R. Swick. Resource Description Framework (RDF) Model and Syntax Specification. W3c recommendation, W3C, February 1999.

[14] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6:167–195, 2015.

[15] X. Li, Y. Xia, X. Long, Z. Li, and S. Li. Exploring text-transformers in aaai 2021 shared task: Covid-19 fake news detection in english. *arXiv preprint arXiv:2101.02359*, 2021.

[16] M. Mayank, S. Sharma, and R. Sharma. Deap-faked: Knowledge graph based approach for fake news detection. *arXiv preprint arXiv:2107.10648*, 2021.

[17] J. Z. Pan, S. Pavlova, C. Li, N. Li, Y. Li, and J. Liu. Content based Fake News Detection Using Knowledge Graphs. In *Proc. of the International Semantic Web Conference (ISWC2018)*, pages 669–683, 2018.

[18] S. Parikh and P. Atrey. Media-rich fake news detection: A survey. 04 2018.

[19] A. Petrova, E. Sherkhonov, B. Cuenca Grau, and I. Horrocks. Entity comparison in rdf graphs. In C. d'Amato, M. Fernandez, V. Tamma, F. Lecue, P. Cudré-Mauroux, J. Sequeda, C. Lange, and J. Heflin, editors, *The Semantic Web – ISWC 2017*, pages 526–541, Cham, 2017. Springer International Publishing.

[20] E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. W3C Recommendation, 2008.

[21] O. Sagi and L. Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.

[22] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu. Fake news detection on social media: A data mining perspective. *SIGKDD Explor. Newsl.*, 19(1):22–36, sep 2017.

[23] O. Software. Virtuoso sparql query editor, 2009.

[24] D. Vrandečić and M. Krötzsch. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, sep 2014.

[25] W. Y. Wang. "liar, liar pants on fire": A new benchmark dataset for fake news detection. *CoRR*, abs/1705.00648, 2017.

[26] C. Wardle. Fake news. it's complicated. *First Draft*, feb 2017.

[27] X. Zhou and R. Zafarani. A survey of fake news: Fundamental theories, detection methods, and opportunities. *ACM Comput. Surv.*, 53(5), sep 2020.