National College of Ireland

An AI Approach to Investigate the Identification of Fake News in Brazilian Portuguese

MSc Research Project
MSc in Data Analytics

Marcelo Fischer
Student ID: 20118872

School of Computing
National College of Ireland

Supervisor: Rejwanul Haque

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Marcelo Fischer
Student ID:	20118872
Programme:	MSc in Data Analytics
Year:	2021
Module:	MSc Research Project
Supervisor:	Rejwanul Haque
Submission Due Date:	16/08/2021
Project Title:	An AI Approach to Investigate the Identification of Fake News in Brazilian Portuguese
Word Count:	5793
Page Count:	17

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	15th September 2021

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

An AI Approach to Investigate the Identification of Fake News in Brazilian Portuguese

Marcelo Fischer
20118872

Contents

List of Figures

List of Tables

Abstract

Spread of fake news is damaging to the society as a whole. Given the amount of news produced each day it is vital to automate the process of fake news identification. However, this is still a challenge specially for languages other than English. This research proposes an AI stacking approach, an ensemble method, three deep neural networks, and mBERT to identify fake news in Brazilian Portuguese. The AI stacking approach combines seven machine learning models to detect fake news in Brazilian Portuguese. The dataset used is the Fake.Br Corpus with 7200 news (3600 real and 3600 fake). Truncated texts were considered to avoid the bias of the length of the texts. The best performing model in this scenario, with the use of term frequency (TF) was the LinearSVC with almost 97% recall, followed by the stacking model with 96.3% recall. The deep learning architectures were not able to outperform the LinearSVC or the stacking model. The mBERT model was able to achieve 99.4% recall. The results shown here extend further the understanding of fake news identification in Brazilian Portuguese.

1 Introduction

Fake news can be described as the act of knowingly or intentionally publishing distorted or false news content online and in the past few years it has dominated the social media (Klein and Wueller; 2017). Even though fake news is not a new term or trend (it roots back to 1600s with the name of "Propaganda" (Gravanis et al.; 2019)), the easy and cheap access to the internet and social media accounts has made this a much bigger problem in the 21st century. In the worst case scenario, where real news and fake news cannot be distinguished, society could possibly find itself in the brick of collapse as it would no longer be possible to hold value in truth.

Some fake news are so convincing that they result in damaging actions towards specific individuals and/or the society. The advance in AI made possible to impersonate other people and damage their public image (Newman; 2021). In 2016, a man threatened people with guns due to fake news (Kang and Goldman; 2016). In addition, fake news have already influenced presidential elections in USA (Vosoughi et al.; 2018).

Researchers use different approaches to identify different types of fake news. *Deliberate Misinformation* are false information that are spread targeting specific users (Kaur et al.; 2020). *Clickbait* uses sensationalist material and titles to attract users to click on the news to generate huge amounts of revenue. Furthermore, fake websites bombard users with ads (Shu et al.; 2017). *Parody or Satirical* uses exaggeration and absurdity to talk about real events and generate a chaotic situation for the readers (Rubin et al.; 2016). *Hoaxes* are fake news posted on purpose to cause harm or prejudice to the reader (Kaur et al.; 2020). *False Headlines* are news with sensationalist headlines to draw the reader's attention. Many times, the content is not linked with the title (Kaur et al.; 2020).

Over the past few years, a lot of effort has been put into developing mechanisms to identify fake news rapidly to avoid its publication and spread. Sharma and Sharma (2019) describe steps to manually identify fake news. However, this is not feasible or practical or cheap and people even use bots to generate fake content constantly on the internet (Zhang and Ghorbani; 2020; Shao et al.; 2017). Another drawback of manually checking is that it does not prevent content from being published, it can only act afterwards. Thus, automated identification mechanisms are of great importance.

Given the challenges, several different ML models have been applied to fake news identification, but a definite solution is not yet available. On top of that, most of the studies are done in the English language. In light of the above, the objectives of this study are to extend previous works done in the Fake.Br Corpus dataset[1] to incorporate ensemble, stacking and deep learning models and see if the performance can be improved. The aim of this research is to investigate to what extent an AI stacking approach and deep learning models can identify fake news in low resource languages such as Brazilian Portuguese and provide a wider range of baseline models for comparison by future works.

The major contribution of this research is to extend previous works done in the Fake.Br corpus dataset to encompass more machine learning models and deep learning models to further investigate the identification of fake news in Brazilian Portuguese and create a better baseline of models for future comparison.

The rest of this report is structured as follows: in section 2 previous works related to fake news detection are discussed. In section 3 the proposed methodology is presented in detail. In section 4 the design and implementation of the models are shown. In section 5 the findings are presented and discussed. Finally, in section 6 the work is concluded and future works are stated.

2 Related Work

AI have evolved in the last decades and a lot of research has been carried out in the field of text analysis. Many people have put together labelled datasets containing fake news articles, tweets, blog posts among other types of texts that are used in the web. Additionally, different datasets are related to different ML tasks. Some use regression to give a score to the truthfulness of a piece of news, but it is not common or widely used. More often, researchers use classification tasks to identify fake news. Some classification tasks are binary, such as belonging to the "true" or "fake" group. However, making the task always binary is tricky given that a lot of texts might be partially true or fake, so it is common to create an intermediate class in which the piece of news is not completely fake or completely true. Moreover, some researchers attempt to determine whether the text of a news article is related or not to its given headline. The most famous dataset for the latter is the Fake News Challenge (FNC-1) dataset[2]. Section 2.1 reviews studies that classify news articles into more than 2 classes (multi-class classification). Section 2.2 reviews studies that have attempted to use binary classification to identify news as fake or real.

2.1 Multi-class Tasks

As stated above, separating fake news into two groups might be tricky since a lot of news are not entirely fake or real. With that in mind, some groups have put together different datasets with different tasks. Regarding classifying social media statements another problem arises. The statements are often small in length (tweets, for example) and therefore datasets with big news texts are not suitable to train models to predict fake social media statements. Vlachos and Riedel (2014) put together one of the first fake news dataset by gathering statements from PolitiFact and Channel 4[3], but the size of it was

[1] https://github.com/roneysco/Fake.br-Corpus

[2] www.fakenewschallenge.org/

[3] www.channel4.com/news/factcheck/

far too small, less than 150 records. Vlachos and Riedel also proposed ML approaches to use but did not implement them. Ferreira and Vlachos (2016) also put together a dataset called "Emergent", where they gathered 300 labeled claims from PolitiFact and each claim belongs to a specific stance relative to the headline. They fitted a logistic regression model focusing on stance classification and were able to improve the accuracy in 26% when compare to other authors. Despite that, both datasets are extremely small when it comes to the quantity of example data to give to a ML algorithm. Because of that, in 2016 Wang (2017) released a new dataset for stance detection called "LIAR". This dataset contains 12.836 short statements got from PolitiFact and each statement can belong to one of six categories: pants-fire, false, barely-true, half-true, mostly-true, and true. The LIAR dataset also provides metadata for the statements such as information about the subject, speaker, party, state, among others. In this work they proposed a hybrid NN model that integrates the text and metadata. Their model is constituted by a convolutional neural network (CNN) with max-pooling and a bidirectional long-short term memory (Bi-LSTM). The CNN deals with the information from the statements and the Bi-LSTM deals with the metadata. The features are then concatenated and passed into a softmax function to classify the statements. The baselines they used were logistic regression (LR), support vector machine (SVM), Bi-LSTM, and a CNN. With the use of word2vec the CNN outperformed all models on the test set.

Goldani et al. (2021) also used the LIAR dataset but implemented a completely new approach. They used capsule neural networks (introduced by Sabour et al. (2017)). This type of network became very famous in computer vision tasks, and after such success researchers started applying it to various different NLP tasks. It is composed of different capsules in which each of them try to predict specific parameters. The same network may contain several capsules and the authors used 3 of them: an n-gram convolutional layer as the first capsule, a convolutional layer and a feed-forward layer, also in a capsule, where the last one made use of the average pooling. The authors used static word embedding in the model. They were able to improve the accuracy of state-of-the-art works by 1%. They also evaluated the model on the ISOT dataset but this will be discussed in section 2.2.

Another common task in the fake news domain is stance detection. The most famous dataset for this task is the FNC-1. This dataset has 75.385 labeled examples and the task is to give the model a headline and a piece of news and then classify it in one of the four stances:

- Agrees: The content concurs with the headline.
- Disagrees: The content diverges from the headline.
- Discusses: The content debates about the same subject as the headline, but does not concur or diverge.
- Unrelated: The content talks about a topic that is not related to the headline.

Umer et al. (2020) used the FNC-1 dataset with basic text preprocessing steps (stopwords removal, converting to lower case, stemming, removing symbols, etc) and then the clean data was vectorized with word2vec. They proposed a hybrid CNN/LSTM neural network and four different data models were fed to it:

1. The raw data was given to the model, without preprocessing or dimensionality reduction.

2. The preprocessed data was fed to the model without using dimensionality reduction.

3. The input was the preprocessed data after using chi-square test for independence for feature selection and dimensionality reduction.

4. The input was the preprocessed data after using principal component analysis (PCA).

The biggest difference for the model proposed by Umer et al. (2020) was applying dimensionality reduction to the data prior to feeding it to the network. With this, they were able to reduce the training time by almost half and when comparing all of the models discussed above the CNN/LSTM in combination with PCA obtained the best result overall with 97.8% accuracy and F1-score in the FNC-1 dataset. The model with the raw data resulted in 78.4% accuracy and 81.9% F1-score. Their model outperformed the state-of-the-art deep learning models such as BERT (Devlin et al.; 2018), XLNet (Yang et al.; 2019) and RoBERTa (Liu et al.; 2019). However, their work is tied to the English language and the use of PCA requires the features to have a degree of correlation in order to work.

2.2 Binary Classification Works

The work proposed here will focus on a binary classification task as these are the most common task carried out on fake news datasets. There are usually two different ways to implement data models for these tasks: text preprocessing considerations such as n-grams, term frequency (TF), term frequency-inverse document frequency (TF-IDF), and linguistic features such as part of speech (POS), word categories and so on. Gravanis et al. (2019) have conducted a thorough study and also put together a novel dataset for fake news identification called the UNBiased dataset. The authors considered that getting news from few sources could lead to a biased model toward those specific sources. Also, the news need to be from different topics to be able to generalize. Another point is that their study is focused on using linguistic features, and the same sources would potentially use the same linguistic features. Thus, they followed the following rules to generate the UNBiased dataset:

- Each piece of news should be labeled by an expert.
- The examples should come from different sources.
- The real news must be released by well known and trustworthy news organizations.
- The examples should be from different topics (economics, politics, ...).

With the dataset in hands, Gravanis et al. (2019) proposed a benchmarking for the features sets suggested by Burgoon et al. (2003), Newman et al. (2003) and Zhou et al. (2004) where single sets and all of their possible combinations together with word2vec word embedding were considered. The full combined feature set with word2vec was the best overall, and it was fed to a Naive Bayes, SVM, DT, KNN along with 2 ensemble methods namely AdaBoost and Bagging. The model was evaluated against 5 different datasets and the best model was the AdaBoost with the feature set mentioned above that achieved 95% accuracy on all datasets. SVM and Bagging were right behind AdaBoost

and presented no statistical significant difference. A possible improvement to their model would be to consider metadata about the news articles such as publisher.

Kaur et al. (2020) used the News Trends, Kaggle and Reuters datasets and implemented a multi-level voting approach to classify the news articles. They tried three different feature extraction techniques, namely TF-IDF, count vectorizer (CV) and hashing-vectorizer (HV) and used common text preprocessing steps such stopwords removal. They fitted a total of 13 different ML algorithms, each belonging to one of the following families: Naïve Bayes, SVM, DT, Passive Aggressive (PA), Stochastic Gradient Descent (SGD), LR, MultiLayer Perceptron (MLP), AdaBoost, Gradient Boosting or Voting. Even though HV is the most memory efficient, it resulted in the lowest performance between the three feature extraction techniques. After evaluating all the models, the best one from each category (CV, HV and TF-IDF) were chosen based on the false positive (FP) rate and separated into groups that complemented each other. These groups were then used as input to the three-level voting model proposed by them. For the Kaggle dataset, their model was able to achieve 98.9% accuracy when using TF-IDF. Since they use a voting mechanism, the model is more efficient due to the possibility of parallel computations. On the other hand, they did not incorporate any linguistic features, and also only used uni-grams. Lastly, the datasets used are limited to 3, and each of them was generated using the same sources.

To identify fake news it is important to understand all of the common characteristics that they share. Xu et al. (2019) profiled the fake news websites' publishers, rankings and popularity and compared to real news websites. While, on average, almost 80% of the fake news publishers prefer to stay anonymous, less than 2% of real news posted are from anonymous publishers. Beyond that, fake news websites almost always have new domains, while trusted domains are usually more than 12 years old. Regarding the rankings, they concluded that credible sources of news are usually at the top and fake news websites quite often do not even appear on the top 1 million. They also found out that credible websites are usually more popular and have much higher page views per visitor. Lastly, they also say that, for the datasets used, while real news can always be found online, fake news usually disappear, where 55% could not be found, on average. Xu et al. (2019) also tried to identify the most relevant terms in fake and real news and their conclusion is that they are very similar, so this is not a good way to classify them. Another attempt was to use Latent Dirichlet allocation (LDA) to identify the topics in each class but they were also similar. As a final attempt they measured the similarity between a chosen article and all of the others, both real and fake, using the Jaccard similarity. The fake news present a strong similarity between them and this could be used to identify them. Xu et al. (2019) used 3 very small datasets and each of them are from a short period of 3 months during the year of 2016. Even though the study shows good results, the examples they used are extremely limited and they could be biased or not representative.

The web is a very fluid environment in which posts are constantly being generated. Fake news are no different and therefore it is important to keep the examples up to date. Elhadad et al. (2020) got data from the WHO, UNICEF, and UN websites as their ground-truth (focused specifically on COVID-19 data) and web-scraped from different fact-checking websites using the Google Fact Check Tools API. The authors collected a lot of textual data in various different languages but kept only English. They also used TF-IDF as the feature extraction technique and tested several models on their data using a voting ensemble and considering from uni-grams to tri-grams and word embeddings.

They evaluated the models using 12 different metrics and the best one overall was the NN which achieved up to 99% F1-score for the COVID-19 related topics and their model could be extended to taking in real-time data. The final product is available here[4].

Jiang et al. (2021) proposed a new stacking approach for fake news detection and tested 9 different algorithms on the ISOT and KDnugget datasets. They performed grid search to optimize the hyper-parameters of the models and used different feature extraction methods such as TF, TF-IDF and word embedding. A major downside of their work is that they have not provided the parameters or the values used in the grid search, so it is hard to reproduce and test their work. After evaluating the methods, they chose the best one overall (RF) and used it to train the predicted data again. As a result they were able to get an accuracy of 99.96% on the ISOT dataset and an accuracy of 96.05% on the KDnugget dataset. Their work is extremely accurate and the authors themselves proposed to test the model in different languages. Once again we can see that the works are exclusive to the English language.

In summary, there are a lot of very thorough studies and well implemented models ranging from simple LR to hybrid neural networks that can give outstanding performances. However, almost all the studies carried out so far in the fake news domain are exclusive to the English language. The work proposed here is to extend the models applied to the Fake.Br dataset and try to achieve performances as high as in the English language. More specifically, the intention of the research question is to help the detection of fake news in Brazilian Portuguese and to widen the current baseline models for future comparison with modern models. Finally, as discussed above, ensemble and stacking models usually outperform simpler models and are also fast to train, thus they were chosen for this research.

3 Methodology

The steps followed to conduct this research are described in this section in the same order as they were applied. The data mining methodology used was Knowledge Discovery in Databases (KDD).

3.1 Dataset

Silva et al. (2020) have put together a novel dataset for fake news detection in the Brazilian Portuguese language called Fake.Br Corpus. The authors followed the guidelines proposed by Hovy and Lavid (2010) and Rubin et al. (2015) to create the corpus. The dataset consists of corresponding true and fake news articles in which the real news denies the fake correspondent or, at least, is related to the same topic. Having pairs of fake and real news helps machine learning algorithms to detect patterns and avoid bias. Silva et al. (2020) used a semi-automatic approach to build the dataset. 3600 fake news were manually selected by the authors, then they implemented a web crawler to collect real news from trusted Portuguese sources. To find similar texts they used keywords from the previously collected fake news to find the corresponding real one. To choose the real news they used cosine lexical similarity to choose the 3600 corresponding real news. As a final step, Silva et al. (2020) also checked the collected real news themselves to assure that they were at least topic related to the paired fake news. It is important to point

[4]https://github.com/mohaddad/COVID-FAKES

out that Silva et al. (2020) excluded any news that were half true and considered only real or fake labels and texts. Additionally, the collected news are from different topics (politics, religion, economy, ...). The final dataset consists of equally distributed 7200 news articles.

3.2 Pre-processing

The Fake.Br corpus consists of real news articles collected from the web. Therefore, a lot of the texts have URLs that carry no useful information in their texts, however the presence of an URL in the text might be a pattern to help distinguishing between fake and real news. URLs were normalized to the string "url". Emails were also in some texts and were normalized to the string "email". Next, words that contained digits were removed. Squared brackets and Unicode characters were also removed. Then, inverted commas and "-" were also removed from the texts. Lastly, any double or trailing spaces were removed. As a final step, any non-alphanumeric characters were removed. After cleaning the data, three different methods were applied to it: TF, TF-IDF and embedding.

The cleaning steps of the pre-processing were performed using the Python re (regular expressions) library. The TF and TF-IDF techniques were implemented using the SciKit Learn library's "CountVectorizer" and "TfidfVectorizer" functions, respectively. For the deep learning models, the "Tokenizer" function from Keras was used.

3.3 Methods

The following models were implemented in this research:

- **Logistic Regression:** One of the simplest and most used algorithms for classification tasks, generally binary classification. It uses the sigmoid function to calculate a probability for each case and, based on a threshold, it assigns the case to one of the two classes.
- **Decision Tree:** It is a common algorithm usually used for classification tasks but can also be used for regression. It is a top-down approach that can be shaped like a tree. It learns by splitting the data into nodes where each node is tied to a class label. There are different methods used for choosing how to split the nodes such as information gain or gain_ratio.
- **Random Forest:** It is an ensemble algorithm that consists of several trees being averaged, which prevents it from overfitting. It diminishes the correlation between the different trees. For each split, the algorithm takes a random subset of the features which dictates the following split. Random forests are widely used for classification and regression tasks, and they provide very good results even when compared to modern neural networks.
- **K-Nearest Neighbor:** It is a very simple, yet powerful algorithm also used for classification and regression tasks. The algorithm calculates the nearest points based on a chosen distance, e.g., Euclidean, Manhattan, etc. Then, it assigns the test sample a label based on a majority voting strategy.
- **Linear SVC:** The model calculates a line/plane/hyperplane that separates the different classes. The hyperplane tries to maximize the distance between points that are from different classes but that are close to each other. However, can only successfully separate data that is linearly separable.

- **Support Vector Machine:** Works in a similar way to the Linear SVC, however, if the data is not linearly separable, SVM is still capable of making classification using the kernel trick, which maps the data points to a higher dimensional space where they can be separate by a plane/hyperplane.
- **Multinomial Naïve Bayes:** Efficient algorithm commonly used for text classification tasks. It is based on the Bayes Theorem of probability and assumes that each feature is independent from one another.
- **XGBoost:** It is a very robust and powerful ensemble technique that also makes use of trees. It generates one classification tree at a time and improves the next one using stochastic gradient boosting. It is very stable, accurate and robust against overfitting.
- **Stacking:** It is another ensemble technique that consists of two layers. The first layer is composed of baseline models that produce predictions on the dataset. The second layer contains a meta-classifier that takes in the baseline models' predictions as inputs and generates new predictions.
- **Convolutional Neural Network:** CNNs are most often used for image classification and computer vision, however they can also be used for sequence text data as they are good at detecting patterns. With the help of word embedding, it is possible to represent each word as a vector in a vector space that captures the words relationships and similarities. In the case of text data, the convolution operation happens in one dimension, the word vectors.
- **Long Short-Term Memory:** One of the most widely used NN for sequence data such as text data and time series. It consists of LSTM cells which contain different gates inside, namely forget gate, input gate and output gate. The forget gate deletes information not deemed important in the long term state. The input gate decides how much of the old information needs to be passed along to the future and the output gate allows data to influence the output of the current time-step. Since the network is able to retain information, it does not suffer from the vanishing gradient problem.
- **Gated Recurrent Unit:** GRUs are a simpler version of LSTMs. In a GRU there only two gates: input gate and forget gate. GRUs and LSTMs came to solve the vanishing gradient problem present in normal recurrent neural networks. Once again, the input gate decides how much of the old information needs to be passed along to the future, while the forget gate is used to decide how much of the old information to forget. Even being simpler than LSTMs, sometimes GRUs can perform better and are faster to train.
- **Multilingual BERT:** it is an algorithm that combines a bidirectional approach with a deep transformer encoder network. It is extremely powerful and can be used in NLP tasks for more than 100 different languages, including Brazilian Portuguese.

3.4 Evaluation Criteria

To compare and evaluate the different models proposed the following metrics were used for all experiments:

- $accuracy = \frac{TP+TN}{TP+TN+FP+FN}$, tells how often the classifier is correct.
- $precision = \frac{TP}{TP+FP}$, tells how often the classifier is correct when it predicts positive..

- $recall = \frac{TP}{TP+FN}$, tells the proportion of actual positives that are correctly identified as such.
- $F1\_score = 2\frac{Prec \times Rec}{Prec+Rec}$, harmonic average of precision and recall.

4 Design & Implementation

This section specifies the system's specifications used to run all experiments, the programming language and libraries' versions, the parameters used for the models and the architectures of the deep neural networks used.

4.1 System Specifications & Versions

The hardware specifications used in all experiments were 32GB of RAM, Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz processor and Ubuntu 20.04 and Windows 10. The Python version and all the libraries' versions are shown in Table 1.

Table 1: Python Libraries and Versions.

Library	***Version***
Python	3.8.5
Jupyter Lab	3.0.14
pandas	1.2.4
numpy	1.19.2
re	2.2.1
tensorflow	2.3.0
keras	2.4.3
scikit learn	0.24.2
nltk	3.6.2

4.2 Models' Designs and Description

This subsection gives the detailed information on the models' parameters and architectures. All models were optimized using grid search but for the XGBoost. Given time constraints, XGBoost was optimized using random search. All searches used the cross-validation parameter equals to 5.

- **Logistic Regression:** values of C analysed were [0.01, 0.1, 1, 10, 100].
- **Decision Tree:** the optimized parameters in the search were:
  - criterion = ['gini', 'entropy']
  - splitter = ['best', 'random']
  - max_features = [None, 'auto', 'log2']
  - max_depth = [None, 2, 3, 4, 5, 6]
  - min_samples_split = [2, 5, 7, 10]
  - min_samples_leaf = [1, 2, 4]
- **Random Forest:** the optimized parameters in the search were:
  - n_estimators = [10, 20, 30, 40, 50, . . . , 150]

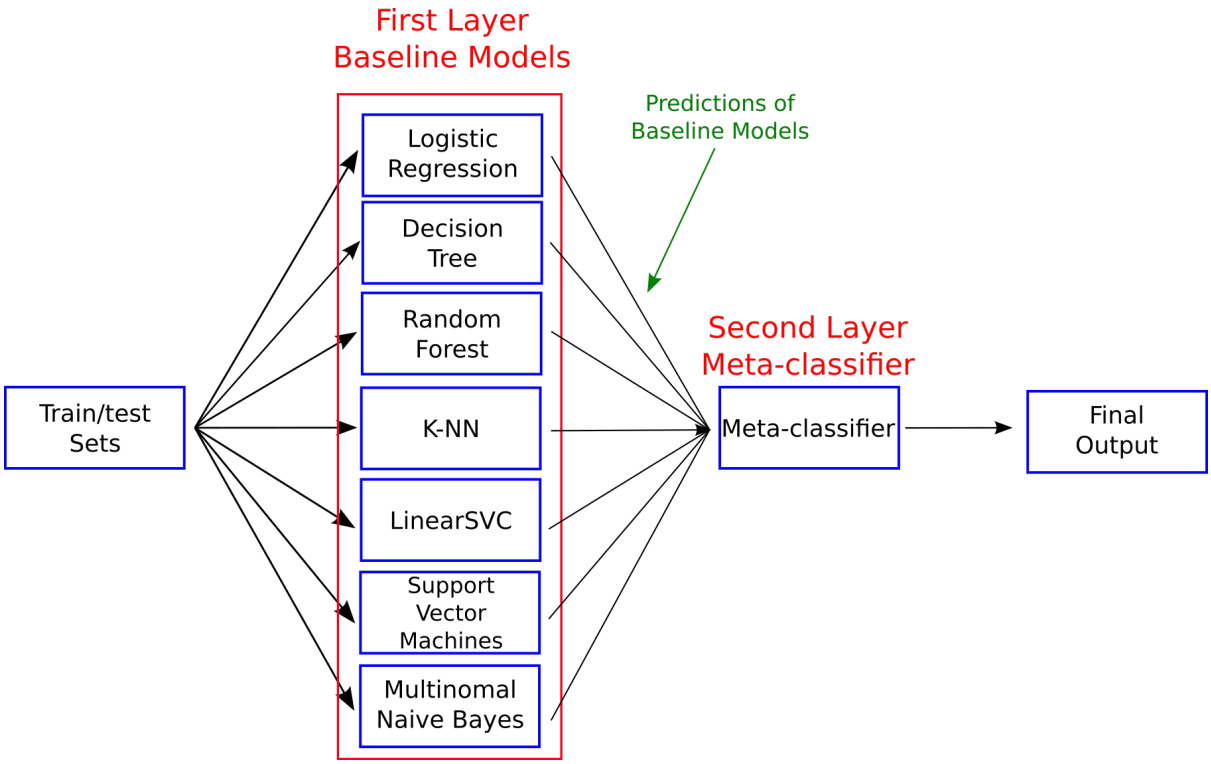
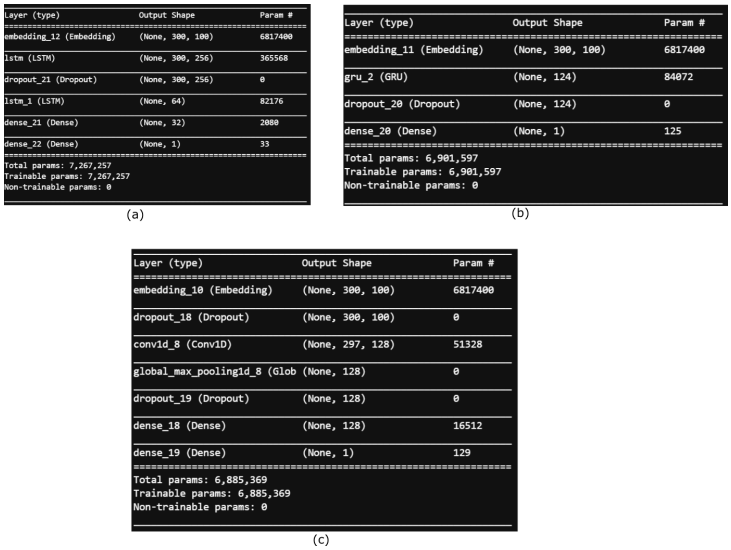


Figure 1: Architecture of the stacking model used.



(a)

Layer (type)	Output Shape	Param #
embedding_12 (Embedding)	(None, 300, 100)	6817400
lstm (LSTM)	(None, 300, 256)	365568
dropout_21 (Dropout)	(None, 300, 256)	0
lstm_1 (LSTM)	(None, 64)	82176
dense_21 (Dense)	(None, 32)	2080
dense_22 (Dense)	(None, 1)	33

Total params: 7,267,257
Trainable params: 7,267,257
Non-trainable params: 0

(b)

Layer (type)	Output Shape	Param #
embedding_11 (Embedding)	(None, 300, 100)	6817400
gru_2 (GRU)	(None, 124)	84072
dropout_20 (Dropout)	(None, 124)	0
dense_20 (Dense)	(None, 1)	125

Total params: 6,901,597
Trainable params: 6,901,597
Non-trainable params: 0

(c)

Layer (type)	Output Shape	Param #
embedding_10 (Embedding)	(None, 300, 100)	6817400
dropout_18 (Dropout)	(None, 300, 100)	0
conv1d_8 (Conv1D)	(None, 297, 128)	51328
global_max_pooling1d_8 (Glob	(None, 128)	0
dropout_19 (Dropout)	(None, 128)	0
dense_18 (Dense)	(None, 128)	16512
dense_19 (Dense)	(None, 1)	129

Total params: 6,885,369
Trainable params: 6,885,369
Non-trainable params: 0

Figure 2: Architectures of the neural networks. (a) is the LSTM, (b) is the GRU and (c) is the CNN.

Figure 3: Architecture of the mBERT model.

  - max_features = ['auto', 'sqrt']
  - max_depth = [1, 2, 4, 8, 16, 32, None]
  - bootstrap = [True, False]
- **K-Nearest Neighbour:** was optimized using the elbow method.
- **LinearSVC:** values of C optimized were [0.001, 0.003, 0.006, 0.01, 0.03, 0.06, 0.1, 0.3, 0.6].
- **Support Vector Classifier:** values of C optimized were the same as LinearSVC and the kernel = ['linear', 'rbf'].
- **Multinomial Naïve Bayes:** values of alpha optimized were [0.0001, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 1.5, 2, 5].
- **XGBoost:** the optimized parameters in the search were:
  - n_estimators = [100, 300, 500, 700]
  - max_depth = [2, 3, 4, 5, 6]
  - learning_rate = [0.01, 0.05, 0.1, 0.3]
  - min_child_weight = [1, 2, 3, 4, 5]
  - booster = ['gbtree', 'gblinear', 'dart']

The architecture for the stacking model can be seen in Figure 1. The meta-classifier used in each experiment is given in section 5. The architectures for the neural networks can be seen in Figure 2. The dropout ratio was 0.3 for all dropout layers. Two callbacks were used while fitting the networks, EarlyStopping and ModelCheckpoint. The first allows the training to stop if no improvement is happening in a chosen metric. The second allows to save the best model among all epochs ran. The architecture for the mBERT model can be seen in Figure 3.

5 Results & Discussion

All experiments involving machine learning models were performed using the Bag-of-Words (BoW) approach. For the deep learning models an embedding layer was added to represent the words as vectors. The true news in the Fake.Br corpus are usually much bigger than the fake (see Table 2) and this is usually what happens in other datasets as well. To check if the text size can influence in the results two different scenarios were tried: full texts and truncated texts.

Label	**Average Length (in characters)**
True	6674
Fake	1124

Table 2: Average Length of Texts for each Class.

5.1 Term Frequency

Table 3 shows the results for the experiments using the ML models and the TF technique. Bold values indicate the best scores. When considering full texts, the best individual classifier overall is the LinearSVC which was able to correctly classify almost 98% of fake news. Therefore, the meta classifier in the stacking model in this scenario was chosen to be the LinearSVC. The stacking model was superior to the individual models with a precision of almost 98% and a F1_score of 97.5%. For the truncated texts, each input was limited to 200 tokens. In this case, the best individual classifier overall was the LinearSVC. Furthermore, the LinearSVC was superior to the stacking model. Here, the meta classifier used for the stacking model was the linear regression, which gave better results for the stacking. What's more, almost all models for the truncated texts had worse performances when compared to full texts. This indicates that the length of the text impacts the results and that taking that into consideration is important. Classification methods that use the size of the texts as features might generated overestimated results. Using other attributes from the documents, i.e., linguistic features, could avoid a biased model and make it more reliable and trustworthy for real applications. In addition, the size of the text is a feature that can be easily controlled by the writer.

In comparison to Silva et al. (2020), the use of TF and truncated texts, without the normalization of IDF, produced better results for the individual models with an increase

Table 3: Metrics for the Experiments using TF and Full/Truncated Texts

	TF - Full Text				*TF - Truncated Text*			
	Accuracy	Precision	Recall	F1_score	Accuracy	Precision	Recall	F1_score
Stacking	**0.9750**	**0.9766**	0.9735	**0.9749**	0.9674	0.9719	0.9624	0.9671
LinearSVC	0.9715	0.9656	0.9777	0.9716	**0.9715**	**0.9734**	0.9694	**0.9714**
LR	0.9715	0.9721	0.9708	0.9714	0.9632	0.9586	0.9680	0.9633
SVC	0.9660	0.9639	0,9680	0.9659	0.9646	0.9638	0.9652	0.9645
RF	0.9542	0.9441	0.9652	0.9545	0.9569	0.9659	0.9471	0.9564
DT	0.9444	0.9506	0.9373	0.9439	0.9438	0.9505	0.9359	0.9432
KNN	0.9319	0.8904	**0.9847**	0.9352	0.9472	0.9224	**0.9763**	0.9486
NB	0.8625	0.9290	0.7841	0.8505	0.7840	0.8246	0.7201	0.7688

of 3.74% in the recall for the best classifier. In some situations, frequently occurring words are a strong indicative of the task in hand. Fake news contain more slang and misspelled words than real news from trustworthy sources. This is true for the Fake.Br corpus (Silva et al.; 2020). Therefore, the use of IDF reduces the importance of such words and thus work against the model. When comparing the stacking approaches, the results for Silva et al. (2020) and for this study show very similar results. This indicates that adding more models to the stacking did not improve the overall stacking results. This shows that the simpler approach proposed by Silva et al. (2020) is better.

5.2 Term Frequency – Inverse Document Frequency

This experiment consisted of applying the extended stacking approach proposed above with the use of TF-IDF. The results can be seen in Table 4.

As stated above, the model's performance is worse when the size of the text is normalized for both real and fake news, nevertheless the model performs very well. However, the use of the IDF normalization caused a drop in the performance of the classifiers. The use of IDF removes the importance of frequently occurring words which can be a strong signal in fake news detection. This signal is lost when the frequencies are normalized. Using only TF yields better results for this study.

Table 4: Metrics for the TF-IDF Stacking Experiments.

	Accuracy	Precision	Recall	F1_score
TF-IDF Stacking Full Text	**0.9681**	**0.9719**	**0.9638**	**0.9678**
TF-IDF Stacking Truncated Text	0.9625	0.9703	0.9540	0.9621

5.3 XGBoost Ensemble

Table 5 shows the results for the XGBoost ensemble model. Once again, the truncated texts show lower accuracy when compared to the full texts. In addition, using only TF as a feature engineering technique also provides better results. The reason is the same as discussed above. The LinearSVC and the stacking model for the truncated texts using TF have better performances than the XGBoost ensemble.

XGBoost is very powerful, however it is highly dependent on hyper-parameter tuning. For this experiment, random search was used to optimize the hyper-parameters. Thus, not all possible combinations of the grid were tried. The lack of computational power available was a constraint for a wider grid and also for the use of grid search instead of random search. This model might still be a reasonable choice if more computing power is available.

5.4 Deep Learning

All experiments here considered truncated texts with 300 tokens and each token was represented by an embedding vector with 100 dimensions. Two different setups were analysed: with and without stopwords. Only the best results which were saved by the ModelCheckpoint callback during training are shown. Table 6 show the results.

Table 5: Metrics for the XGBoost Experiments.

	Accuracy	Precision	Recall	F1_score
TF XGBoost - Full Text	**0.9667**	0.9745	**0.9582**	**0.9663**
TF-IDF XGBoost - Full Text	0.9667	**0.9772**	0.9554	0.9662
TF XGBoost – Truncated Text	0.9618	0.9662	0.9568	0.9615
TF-IDF XGBoost - Truncated Text	0.9562	0.9672	0.9443	0.9556

Table 6: Metrics for the NN experiments.

	Without Stopwords				With Stopwords			
	Accuracy	Precision	Recall	F1_score	Accuracy	Precision	Recall	F1_score
CNN	**0.9514**	**0.9513**	0.9513	**0.9513**	**0.9188**	0.9574	**0.8760**	**0.9149**
GRU	0.9368	0.9219	0.9540	0.9377	0.9118	0.9668	0.8524	0.9060
LSTM	0.9278	0.9008	**0.9610**	0.9299	0.9139	**0.9685**	0.8552	0.9083

The results for the DNNs go against the findings of Silva et al. (2020). The removal of stopwords make the prediction power of the networks increase. This might be due to the fact that neural networks are more powerful in learning and finding patterns in complex data (Mandical et al.; 2020). Thus, removing noise such as stopwords enables the DNNs to have a better performance. However, even the best DNN for this dataset, CNN, was not able to perform as well as the best individual ML models or the stacking model. The best CNN stayed below 96% recall and F1_score. This does not mean that DNNs are worse the ML for this dataset, it is just an indication that the architectures tried here are not optimal and other configurations should be tried.

5.5 Multilingual BERT

The experiment using Multilingual BERT (mBERT) is shown here. Only one setup was analyzed here, using truncated texts with 128 tokens each and without removing stopwords. As can be seen in Table 7, this state-of-the-art algorithm outperformed all the previous ones and was able to achieve a recall of 99.4%. The mBERT model also had the highest F1_score and accuracy when compared to the other models.

Table 7: Metrics for the mBERT experiment.

	Accuracy	Precision	Recall	F1_score
mBERT	**0.9840**	**0.9750**	**0.9940**	**0.9840**

6 Conclusion and Future Work

This research conducted a comprehensive investigation of the identification of fake news in Brazilian Portuguese in the Fake.Br dataset. Several different machine learning models, including stacking and ensemble, as well as deep learning models were fitted to the data. The reported results show that removing the bias of the text length from the models

make the performance drop, but this prevents the algorithm from being tricked by big fake news. Also, the use of TF makes the results improve when compared to the use of TF-IDF. This indicates that fake news have frequently used words that work as signals to the models and shrinking their importance with the IDF normalization works against the models. What's more, the deep learning architectures implemented here are not able to perform as well as the machine learning models with the exception of the mBERT. The same is true for the XGBoost ensemble model. The best performing model in the case of truncated texts and the TF technique was the LinearSVC, followed by the stacking model. Regarding the deep learning models, the removal of stopwords increase their prediction power which is the opposite when compared to the ML models. Finally, the transformer model mBERT without any tuning gave the best results overall and it is the model that the authors recommend using for real applications since it showed more than 99% recall.

It is possible to extend this work further by:

- Performing a wider grid search for the XGBoost model.
- Trying different deep learning architectures.
- Using the combination of different deep neural networks layers.
- Using a stacking model that includes both machine learning and deep learning models.
- Fine tuning mBERT.

Altogether, the results indicate that common machine learning and deep learning models perform well with a recall of almost 97% and an F1_score of 97.1%. However, the state-of-the-art mBERT model has a recall of more than 99% and can possibly be improved with tuning.

References

Burgoon, J. K., Blair, J. P., Qin, T. and Nunamaker, J. F. (2003). Detecting deception through linguistic analysis, *International Conference on Intelligence and Security Informatics*, Springer, pp. 91–101.

Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding, *arXiv preprint arXiv:1810.04805* .

Elhadad, M. K., Li, K. F. and Gebali, F. (2020). Detecting misleading information on covid-19, *Ieee Access* **8**: 165201–165215.

Ferreira, W. and Vlachos, A. (2016). Emergent: a novel data-set for stance classification, *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pp. 1163–1168.

Goldani, M. H., Momtazi, S. and Safabakhsh, R. (2021). Detecting fake news with capsule neural networks, *Applied Soft Computing* **101**: 106991.

Gravanis, G., Vakali, A., Diamantaras, K. and Karadais, P. (2019). Behind the cues: A benchmarking study for fake news detection, *Expert Systems with Applications* **128**: 201–213.

Hovy, E. and Lavid, J. (2010). Towards a 'science'of corpus annotation: a new methodological challenge for corpus linguistics, *International journal of translation* **22**(1): 13–36.

Jiang, T., Li, J. P., Haq, A. U., Saboor, A. and Ali, A. (2021). A novel stacking approach for accurate detection of fake news, *IEEE Access* **9**: 22626–22639.

Kang, C. and Goldman, A. (2016). In washington pizzeria attack, fake news brought real guns, *New York Times* **5**.

Kaur, S., Kumar, P. and Kumaraguru, P. (2020). Automating fake news detection system using multi-level voting model, *Soft Computing* **24**(12): 9049–9069.

Klein, D. and Wueller, J. (2017). Fake news: A legal perspective, *Journal of Internet Law (Apr. 2017)* .

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach, *arXiv preprint arXiv:1907.11692* .

Mandical, R. R., Mamatha, N., Shivakumar, N., Monica, R. and Krishna, A. (2020). Identification of fake news using machine learning, *2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, IEEE, pp. 1–6.

Newman, M. L., Pennebaker, J. W., Berry, D. S. and Richards, J. M. (2003). Lying words: Predicting deception from linguistic styles, *Personality and social psychology bulletin* **29**(5): 665–675.

Newman, N. (2021). Journalism, media and technology trends and predictions 2021.

Rubin, V. L., Chen, Y. and Conroy, N. K. (2015). Deception detection for news: three types of fakes, *Proceedings of the Association for Information Science and Technology* **52**(1): 1–4.

Rubin, V. L., Conroy, N., Chen, Y. and Cornwell, S. (2016). Fake news or truth? using satirical cues to detect potentially misleading news, *Proceedings of the second workshop on computational approaches to deception detection*, pp. 7–17.

Sabour, S., Frosst, N. and Hinton, G. E. (2017). Dynamic routing between capsules, *arXiv preprint arXiv:1710.09829* .

Shao, C., Ciampaglia, G. L., Varol, O., Flammini, A. and Menczer, F. (2017). The spread of fake news by social bots, *arXiv preprint arXiv:1707.07592* **96**: 104.

Sharma, S. and Sharma, D. K. (2019). Fake news detection: A long way to go, *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*, pp. 816–821.

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

Silva, R. M., Santos, R. L., Almeida, T. A. and Pardo, T. A. (2020). Towards automatically filtering fake news in portuguese, *Expert Systems with Applications* **146**: 113199.

Umer, M., Imtiaz, Z., Ullah, S., Mehmood, A., Choi, G. S. and On, B.-W. (2020). Fake news stance detection using deep learning architecture (cnn-lstm), *IEEE Access* **8**: 156695–156706.

Vlachos, A. and Riedel, S. (2014). Fact checking: Task definition and dataset construction, *Proceedings of the ACL 2014 workshop on language technologies and computational social science*, pp. 18–22.

Vosoughi, S., Roy, D. and Aral, S. (2018). The spread of true and false news online, *Science* **359**(6380): 1146–1151.

Wang, W. Y. (2017). " liar, liar pants on fire": A new benchmark dataset for fake news detection, *arXiv preprint arXiv:1705.00648* .

Xu, K., Wang, F., Wang, H. and Yang, B. (2019). Detecting fake news over online social media via domain reputations and content understanding, *Tsinghua Science and Technology* **25**(1): 20–27.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding, *arXiv preprint arXiv:1906.08237* .

Zhang, X. and Ghorbani, A. A. (2020). An overview of online fake news: Characterization, detection, and discussion, *Information Processing & Management* **57**(2): 102025.

Zhou, L., Burgoon, J. K., Nunamaker, J. F. and Twitchell, D. (2004). Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated communications, *Group decision and negotiation* **13**(1): 81–106.