Fake News Outlet Detection

A Comparison of Naïve Bayes versus ANN with BERT

Course name: Natural Language Processing and Text Analytics
Programme: MSc in Business Administration and Data Science

Examination type: Group project

Student names: Chiara Fiorio (149874), Kalina Emilova Georgakieva (149878),

Faye Sabine Hahn (149875), Luca Ludwig (149890)
Supervisor: Rajani Singh, Daniel Hardt

Date of Submission: 30.05.2022

Characters (incl. spaces): 33.143
Number of pages: 15

Abstract

Within the last years, the Internet and especially social media have become a major channel for information retrieval. Unfortunately, the amount of inaccurate and manipulated information is constantly increasing. Many examples for the negative effects of Fake News on our society underline the urgency for solutions. In order to be able to distinguish news articles stemming from credible and Fake News outlets, a dataset from the University of Virginia was used to train two different models. One is a Naïve Bayes classifier, the other one an Artificial Neural Network with an underlying BERT encoder. The Naïve Base classifier alone achieved test accuracy results of over 97%. When stop words were not filtered out in the pre-processing step, the classifier performed even better. This indicates that the filler words are a relevant feature for the classification of fake news. The ANN with BERT achieved an almost perfect test accuracy of 99.98%. Applying the models and comparing the results allows the sector to understand how to detect and flag articles from Fake News outlets as efficiently as possible. With the rise of Fake News and numerous organizations trying to curb its spread, effective application of current technologies is decisive.

Table of Contents

Table of Figures

1 Introduction

1.1 Motivation

The Internet community grows rapidly and so does the speed of information exchange.[1] Within the last years, the Internet and especially social media have become a major channel for information retrieval. Online content plays a significant role in influencing users' decisions and opinions. Unfortunately, the amount of inaccurate and manipulated information is constantly increasing. Many examples for the negative effects of Fake News on our society underline the urgency for solutions. For example, the U.S. presidential elections in 2016 and the Brexit might have been influenced by Fake News and more recently, Fake News about the COVID-19 virus led to an increase of chloroquine drug overdoses and panic buying of groceries and paper products in various countries.[2] Fake News usually aim to influence users' opinions by manipulating the textual and multimedia content. Although this dilemma is not a new concept, the detection of Fake News is believed to be complex given that humans tend to believe misleading information and the lack of control of fake content spread.[3] Platforms like Politifact or GossipCop aim to raise awareness about misinformation posted online. The examination of the articles on those websites is done manually by experts who analyse the content of the articles and determine whether it is fake or not. Still, this task is time consuming and the increasing data volumes overwhelm human abilities. Therefore, the development of an automated system for Fake News detection is a necessity. In this field, Natural Language Processing (NLP) techniques, Machine Learning (ML) and Artificial Neural Networks (ANN) have proven useful. Hence our study aims to contribute to recent literature in the field of automatised Fake News detection.

1.2 Research Question

We aim to extend the existing research within the field of automated Fake News detection by testing different NLP, ML and NN approaches. We focus on textual content in news articles and derive the following research question: Do the structure and features of our textual data provide enough information for a computer system to accurately predict whether an article stems from a Fake News outlet? To answer this question, we exclusively focus on the data's features and structure. We conduct different analyses including N-Gram Analysis, Part-Of-Speech (POS) Tagging, t-distributed Stochastic Neighbor Embedding (t-SNE) Clustering, and Sentiment Analysis to get a profound understanding of our dataset. After that, we apply a Naive Bayes classifier to test the performance of a basic ML algorithm in predicting the correct (true/fake) labels based on the textual data. Finally, we apply and evaluate a NN which is known as one of the most prolific recent advances in natural language processing: A Bidirectional Encoder Representations from Transformers

[1] Oshikawa, Qian & Wang. (2018).

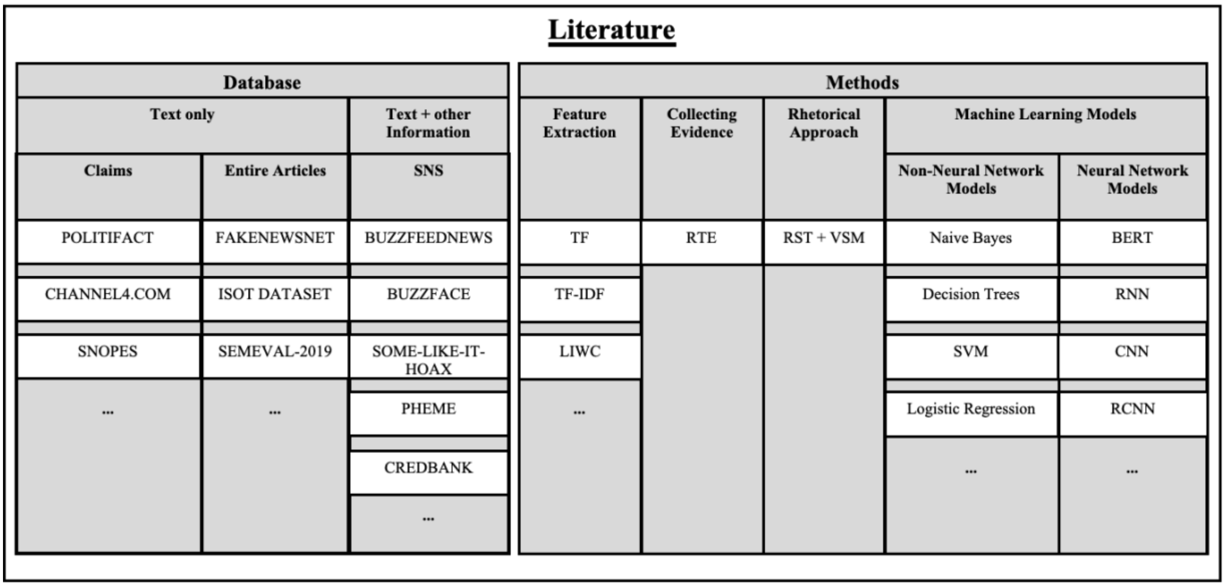
[2] Bovet & Makse. (2019).

[3] Lemann. (2017).

(BERT). All these methods are specifically explained in section 3 and the concrete application is laid down in section 4.

2. Literature Overview

Potential threats of Fake News have led to the development of various countermeasures, some proposed and integrated by social media platforms themselves.[4] Especially within the last decade, many researchers elaborated different automated systems for Fake News detection. In this section, we introduce a framework to give a structured overview of the research landscape for Fake News detection. This framework builds on the work of Oshikawa, Qian and Wang but includes more recent studies as well as a further distinction between different types of databases (see Figure 1).



Literature

Database			Methods				
Text only		Text + other Information	Feature Extraction	Collecting Evidence	Rhetorical Approach	Machine Learning Models	
Claims	Entire Articles	SNS				Non-Neural Network Models	Neural Network Models
POLITIFACT	FAKENEWSNET	BUZZFEEDNEWS	TF	RTE	RST + VSM	Naive Bayes	BERT
CHANNEL4.COM	ISOT DATASET	BUZZFACE	TF-IDF			Decision Trees	RNN
SNOPES	SEMEVAL-2019	SOME-LIKE-IT-HOAX	LIWC			SVM	CNN
...	...	PHEME	...			Logistic Regression	RCNN
		CREDBANK				...	...
		...					

Figure 1 Literature Framework

Research in the field of Fake News detection can be clustered in terms of the database and the methodology that have been used. Regarding the databases, a first distinction can be made between studies that refer to text data only, and studies that take additional information and/or image data into account. While most of the papers only take textual data into consideration, others base their analysis on Social Media Posts (SNS), which are similar to short claims in length but featured by structured data of accounts and posts, including a lot of non-text data.[5] Within the field of pure textual data, a further distinction can be made: some papers focus on claims, whereas others take full news articles as datapoints. Politifact, Channel4.com[2], and Snopes[3] are three sources for short claims in news. Vlachos and Riedel[6] released the first public Fake News detection dataset gathering

[4] Lazer et al. (2018).

[5] Oshikawa, Qian, Wang. (2018).

[6] Vlachos & Riedel. (2014).

data from Politifact and Channel4.com.[7] When it comes to entire articles, Fakenewsnet[8] and Isot[9,10] represent well-established examples. In contrast to that, Buzzfeednews[11] and Some-like-it-hoax[12] are examples for SNS datasets, both focussing on Facebook postings, whereas Pheme (Zubiaga et al., 2016) and Credbank (Mitra and Gilbert, 2015) are two examples for SNS-Twitter datasets. Such SNS datasets do not only include textual data but also additional information including image data. The work of Giachanou, Zhang, and Rosso[13] or Madhusudhan, Mahurkar, and Nagarajan[14] is an exemplary approach that takes advantage of such information retrieved from image data.

Apart from that, studies differ significantly regarding the methods used for classifying the input data into true and fake news. In order to filter out irrelevant and redundant features, Feature Extraction (FE) methods, such as Term Frequency (TF), Term Frequency-Inverted Document Frequency (TF-IDF) and Linguistic Inquiry and Word Count (LIWC) are applied by many researchers for Fake News detection.[15] Most studies perform FE as a pre-processing step before training specific models for classification purposes. Hakak et al. use feature extraction before applying their ensemble model comprising Decision Tree, Random Forest, and Extra Tree classifier. Goldani, Momtazi, and Safabakhsh, R. apply FE before training a capsule neural network and Ahmed, Traore, and Saad take advantage of FE techniques before training six different machine learning classification techniques. Moreover, studies like Dagan et al. [16] detect Fake News by Collecting Evidence with Recognizing Textual Entailment (RTE). Relationships between sentences in a corpus are analysed regarding their accordance with the input data source.[17] Other studies, like Della Vedova et al. [18] take a rhetorical approach, making use of Rhetorical Structure Theory (RST) combined with a Vector Space Model (VSM). RST defines the semantic role of text units, identifies the essential idea, and analyses the characteristics of the input data to detect if the input is fake news or not. VSM then converts texts into vectors and compare them to the centre of True News and Fake News in the RST space.[19]

Lastly, there is a big variety of ML Models and NN used in recent studies. While some studies focus on Non-Neural Network models (Non-NN), others combine or compare them with NN solutions. Non-NN Models include, for example, Support Vector Machine (SVM), Naive Bayes Classifier (NBC), Logistic Regression

[7] Oshikawa, Qian, Wang. (2018).

[8] Shu et al. (2018).

[9] Ahmed, Traore & Saad. (2018).

[10] Ahmed, Traore & Saad. (2017).

[11] Potthast et al. (2017).

[12] Tacchini et al. (2017)

[13] Giachanou, Zhang & Rosso. (2020)

[14] Madhusudhan, Mahurkar & Nagarajan. (2020).

[15] Ahmed, Traore & Saad. (2017).

[16] Dagan et al. (2010).

[17] Oshikawa, Qian & Wang. (2018).

[18] Della Vedova et al. (2018).

[19] Oshikawa, Qian & Wang. (2018).

(LR) and Random Forest Classifier (RFC). Two studies that only use such Non-NN techniques are Erşahin et al. and Granik and Mesyura. Erşahin et al. use Entropy Minimization Discretization on numerical features and analyse the results of the Naive Bayes (NB) algorithm, while Granik and Mesyura use a simple NBC. Meanwhile, Palić et al.[20], apply an SVM approach. Moreover, various studies apply NN strategies to detect Fake News. Earlier studies, apply Recurrent Neural Networks (RNN) (for example Long Short-Term Memory)[21] or Convolutional Neural Networks (CNN)[22], whereas recent literature centres on the application of Bidirectional Encoder Representations from Transformers (BERT). Among the studies that apply BERT, different versions of the network are used: Farokhian, Rafe and Veisi[23] apply MWPBert, which uses two parallel BERT networks to perform veracity detection on full-text news articles. Pandey[24] uses redBERT for the classification, while Szczepański et al.[25] use DistilBERT. Finally, Kula, Choraś and Kozik[26] present a hybrid architecture connecting the BERT network with an RNN.

3 Conceptual Framework

3.1 POS tagging

Part-of-speech (POS) tagging is a process that assigns a special type of label, called POS tag, to a given word in a dataset. The POS tag describes the grammatical relationship between neighbouring words. As such, depending on the language, the types of POS tags can be different. However, for English, the tags could be broadly placed into two categories: closed class and open class. The closed class are usually function words like *a, it, could, and,* which in most cases do not bring any contextual meaning to the sentence, but add grammatical structure.[27] It is referred to as closed class as very rarely a new word could possibly be added. In comparison, the open class POS tags continuously change as it includes the major grammatical structures – nouns, adjectives, and verbs.

3.2 t-SNE

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a projection algorithm used for visualizing high-dimensional data onto a lower-dimensional space.[28] Typically, this is a 2-dimensional scatter plot, plotted in such a way that the maximum of the variance of the original clustering is preserved. t-SNE is a non-linear model, meaning it adapts to the underlying data by performing different calculations on different clustered regions. Initially, the algorithm calculates the conditional probability of how likely it is that two data points

[20] Palić et al. (2019)
[21] Long et al. (2017).
[22] Kim. (2014).
[23] Farokhian, Rafe & Veisi. (2022).
[24] Pandey. (2021).
[25] Szczepański et al. (2021).
[26] Kula, Choraś & Kozik. (2019).
[27] Jurafsky & Martin. (2022).
[28] Van der Maaten & Hinton. (2008).

are neighbors utilizing normal distributions. Then, it places the data points on the lower dimensionality space and calculates their joined probabilities using t-distribution. Finally, using gradient descent, the Kullback-Leibler divergence of the joint probabilities and the conditional probabilities is minimized. It should be noted, however, that t-SNE is computationally expensive, and it could result in different embeddings as the algorithm is stochastic. A way to optimize the algorithm is to either fine-tune the algorithm's hyperparameters or apply dimensionality reduction algorithm, such as Principal Component Analysis (PCA) or Truncated Singular Value Decomposition (SVD), to the data prior to feeding it to the t-SNE.

3.3 Sentiment Analysis

Sentiment Analysis is an NLP technique used for detecting, extracting and analysing data in order to determine the emotion and feelings that the author has towards the discussed topic.[29] Usually, it categorises the information as positive, negative, or neutral, which could provide valuable insights about the data and potentially result in better analysis and classification. As such, it is a broadly studied topic that could be divided into three categories depending on the level of granularity: document level, sentence level and word level Sentiment Analysis.[30] Another way to categorize it is based on the technical approach used where it could be machine-learning based, lexicon-based, statistical and/or rule-based.

An example of a systematic approach for Sentiment Analysis is Valence Aware Dictionary for sentiment Reasoning (VADER), developed by Hutto & Gilbert.[31] It combines quantitative and qualitative methods with five grammatical and syntactical rules to produce a sentiment score on a blog-like text.

3.4 Naïve Bayes

Naïve Bayes (NB) is a supervised ML algorithm, based on the Bayes' theorem, that makes a simplified assumption about the relationship between the features in the dataset.[32] It is a probabilistic classifier, meaning that it estimates the possibility for a given document d to belong to a class $\hat{c}$ using its posterior probability. As such, the most probable class is the one with the highest posterior probability. The probability is calculated based on the product between the prior probability of class c, denoted as $P(c)$ and the likelihood of document d to belong to the class c, denoted as $P(c|d)$. In mathematical terms, the equation looks like the following:

$$\hat{c} = \operatorname{argmax} \quad \underbrace{P(c)}_{prior\ probability} \quad \underbrace{P(d|c)}_{likelihood}$$

However, the computation of all possible combinations is computationally expensive, especially if the set of documents is considerably large. Therefore, simplifying assumptions need to be made to reduce the number of parameters and make the calculations computationally feasible.

[29] Feldman. (2013).

[30] Zhang, Wang, & Liu. (2018).

[31] Hutto & Gilbert. (2014, May).

[32] Jurafsky, & Martin. (2022).

Firstly, NB follows the “bag of word” approach where the position of the word does not matter and only the frequency of occurrence is considered. Intuitively, this leads to the second assumption, known as the naïve assumption, that denotes that the features are independent to one another. Thus, the resulting equation for Naïve Bayes Classifier is the following:

$$\hat{c} = argmax\ P(c) \prod_{f \in F} P(f|c)$$

where f denotes the features of the document.

3.5 BERT

Bidirectional Encoder Representations from Transformers (BERT) is a state-of-the-art pre-trained word embeddings model that unlike previous models, considers context[33]. This means that instead of representing each unique word in a text corpus as a single word vector, it depicts the word as a different vector depending on the context it appears in. This is achieved through transformer encoders that take the whole sequence of tokens as an input at the same time. After taking the original tokens as inputs, BERT adds additional information to the sequences before processing. It adds token embeddings that indicate the beginning and end of a sentence as well as segment and positional embeddings. The former shows to which sentence the token does belong to whereas the latter represents the position of the token within the sentence.

Additionally, the original approach to train BERT is by using masked language model (MLM) that randomly hides words in a text and attempts to make predictions based on the given context. However, as some NLP tasks like question answering require the discovery of relations between pairs of sentences, BERT introduces the next sequence prediction (NSP). In NSP, the model tries to predict whether the next sentence is related to the previous one.

In terms of architecture, the base BERT pre-trained model includes 12 stacked encoders with total of 110M trainable parameters that output 768-dimensional word embeddings. This model can later be fine-tuned for a classification task.

4 Methodology

4.1 Dataset Description

The selected dataset consists of classified news articles stemming from the engineering department of the University of Victoria. The articles labelled as true (“True News”) were scraped from Reuters, while the fake articles come from websites flagged as Fake News resources by Politifact and Wikipedia. This also includes various social media outputs. The key feature for this project is the classification of the news texts as true or fake, but other features are available. On one hand, there is a categorization of subjects (nominal), on the other

[33] Devlin et. al., (2018).

hand the date of publication of the article is available (discrete). In total, the data set consists of 44,898 articles. Of these, 21,417 consist of True News articles, while 23,481 articles belong to the Fake News category. Without duplicates, there are 21,192 True News and 17,455 Fake News articles.

'News', 'politics', 'government news', 'left-news', and 'US_News' are the subjects of the Fake News articles, while the true articles are merely labeled 'politicsNews' and 'worldnews'. For the reason of uneven distribution of subjects, the focus in further analyses was not placed on this category. Furthermore, it is visible that the average length is about 1500 words per article, but the category of Fake News articles contains much more outliers upwards (see Appendix 1).

4.2 Data Pre-processing

Before the data is analyzed, it first needs to be pre-processed. In a preliminary step, duplicates were already removed from the data to retain only unique text instances. The actual Pre-processing phase consists of three main operations, that is stop word removal, tokenization and lemmatization. In each step, one or more new variants of the text corpus get generated. This is due to the decision to use different versions of the dataset for our Analysis which will be covered in detail in the next section.

The first data variant that is generated in the Pre-processing was the “destopped data”. It adjusts the corpus by removing all words that typically have only little or no meaning for the overall sentence, commonly also referred to as “stopwords”. This is done by utilizing *Preprocessing* module from the *Gensim* Python library.

Next, four tokenized variants of the datasets are generated in total. Two of them are derived from the unprocessed data, while the other two build on the destopped data. For both of them, one version gets tokenized on sentence level (“sent-tokenized data”) and one on word level (“word-tokenized data”). Tokenization thereby refers to the process of breaking down one long text string into a sequence of shorter string tokens that – according to the respective tokenization level – each consist of a single sentence or word from the original text string. To do this, we use the tokenizers provided in the *NLTK* Python library. In addition to the standard word and sentence tokenizers, we decided to use the *TweetTokenizer* for the Fake News data as a large portion of it was scraped from Twitter or related social media platforms. This class provides among other things additional features for the removal of Twitter handles (e.g. “@therealdonaldtrump”) and the replacement of repeated character sequences of a length longer than three.

The final step in our Pre-processing is the lemmatization of the textual data. Lemmatization is a text normalization technique to convert each word into its base root mode to allow for easier grouping of words. Thus, by applying lemmatization, the words “runs”, ”running”, ”ran” would all be brought back to the root word “run” and could then be treated as equal which can be helpful for certain NLP tasks. For the means of this project, we use the *NLTK WordNetLemmatizer* to create two lemmatized datasets on top of each of the word-tokenized datasets to obtain one version with and one version without stopwords. Ultimately, this results in seven altered versions of our original dataset. The overall Pre-processing process is illustrated on an example in Appendix 2.

Additionally, it is to be noted that the articles also get transformed to lower case for most of the further analyses and classifications. But as this is usually done automatically through the utilization of vectorizers in our analyses and classifications, we decided not to include it in the Pre-processing process.

Furthermore, since the True News dataset contains 3737 more articles than the Fake News dataset, this class was downsampled to 17455 articles for the classification which is depicted in section 4.5. This is done in order to obtain balanced data across the classes which helps to mitigate the risk of a biased classification.

4.3 Analysis

The goal of this analysis is to come up with a first assessment whether it is feasible to differentiate between articles labelled as fake and as true and which features could potentially be leveraged for their classification. The analysis discusses four different techniques that should serve to deepen the understanding of the dataset and the two classes. Descriptions of the main concepts behind the utilized methods can be found in section 3.

Most frequent N-grams

The first step of the analysis focuses on the most prominent N-grams per class. Therefore, we calculate the average frequency of all Unigrams, Bigrams and Trigrams across each class corpus by utilizing *Scikit-learn*'s *TfidfVectorizer*. Originally, this was done for the original data both, before and after removing stopwords, but as the analysis of the dataset which still contained stopwords did not produce sufficient insights, we will only focus on the analysis of the destopped data. All three variants of the analysis provide different insights. First of all, we see that most N-grams appearing in the list are closely related to U.S. politics and Donald Trump as the main actor in it. In general, the True News data seems to be slightly more balanced than the Fake News data in regard to the average N-gram frequencies but also show similar trends regarding the selection of high frequency N-grams. Furthermore, the analysis highlights again that a lot of the Fake News data seems to be taken from Twitter or is at least referring to its content. This is derived from the fact that N-grams related to Twitter appear across all three variants of the analysis for the Fake News data, while there is no high frequent occurrence at all for the True News data. This difference can be exploited for the means of the Classification task, but also bears the risk of a biased classification for any news content stemming from Twitter. All plots of this analysis step for the destopped data can be found in Appendix 3-5.

POS tag frequency

The second analysis technique that is applied is POS tagging. For the sake of this analysis, we follow a similar approach as it was done for the N-gram analysis, meaning the focus is on the average frequency of each POS tag which are determined with the aid of a language module from *spaCy*. Afterwards, the tags are converted to strings and again handed to the *TfidfVectorizer* to obtain the average frequency of each tag. By this, we want to examine whether there is any difference between the way Fake News and True News are written from a structural perspective. This time, we use the original data (with stopwords) as we want to keep as much of the original text structure as possible. A visualization of the results can be found in Figure 2.

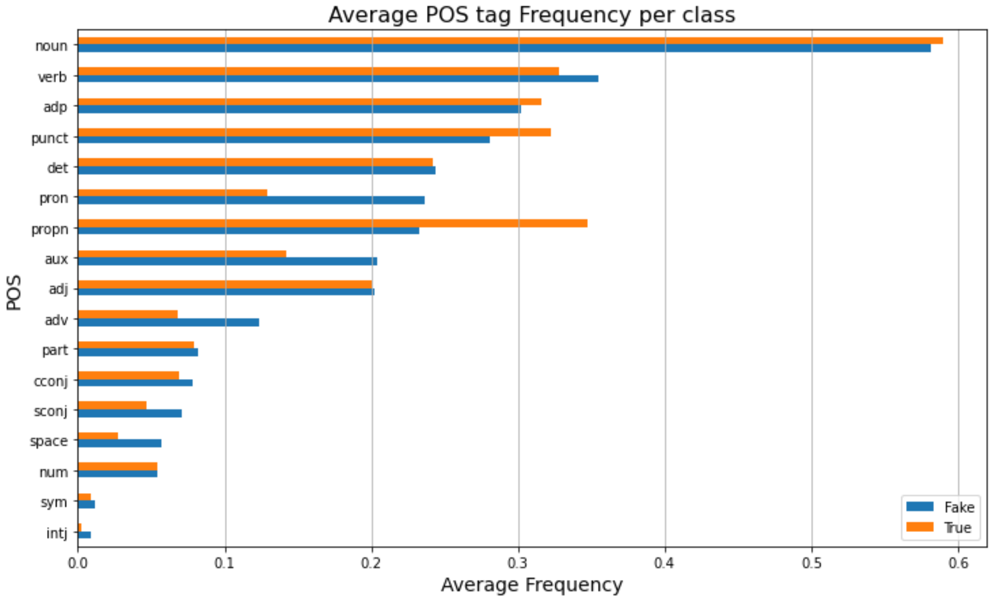


Figure 2 Average POS tag Frequency per class

The most striking results from this analysis suggest that proper nouns (POS tag “propn”) are used notably less in Fake News then they are used in True News. At the same time, they seem to include more pronouns (POS tag “pron”) than True News. Apart from that, there is also a noticeable trend of having more auxiliaries (POS tag “aux”) and adverbs (POS tag “adv”) in Fake News. Altogether, the results from this analysis step imply that there might be some potential to distinguish between True News and Fake News based on their sentence structure and that POS tagging could therefore be leveraged by means of a classification algorithm.

t-SNE Clustering

After focusing on the lexical and structural features separately, applying t-SNE shall bring both characteristics together to see whether there is a clear boundary observable between the clusters of the two classes. The clustering is applied to the lemmatized data, again before and after removing stopwords. Here, we use a combination of truncated SVD and t-SNE, both provided by the *SciKit-learn* library. The resulting plots for can be found in Figure 3.

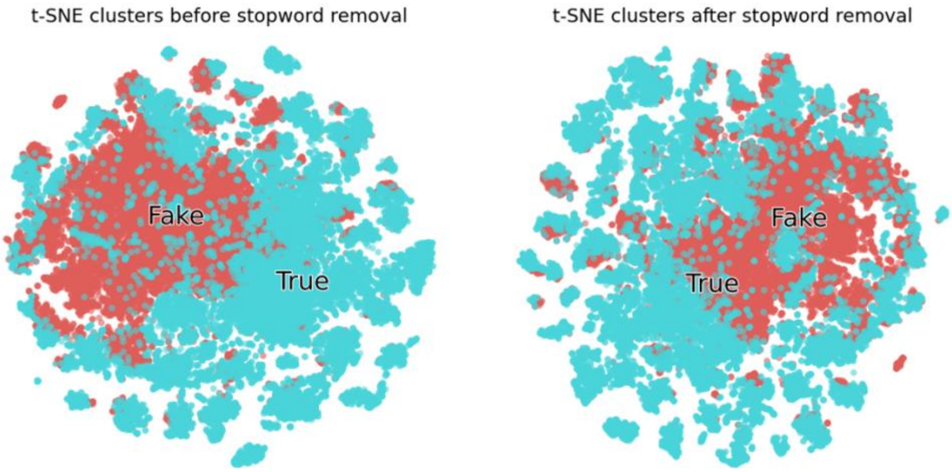


Figure 3 t-SNE clusters before and after stopword removal

In both plots, the clusters show a big overlap and there is no clear boundary between the classes. Nevertheless, there are areas that can be assigned more clearly to either of the two classes. This suggests that a classification of news into Fake News and True News can work to a certain degree, especially if they fall in the less-overlapping areas. Furthermore, it is interesting to observe that the class separation is slightly clearer for the dataset that still includes stopwords. Opposite to the usual approach, this suggests that a Fake News detection could potentially work better without having stopwords removed first.

Sentiment Analysis

Finally, a last analysis step is dedicated to sentiments in the class specific data. This is applied to the sent-tokenized data. We make sure to remove stopwords to avoid the data being biased towards neutrality due to the high number of typically neutral stopwords. We use *NLTK*'s *SentimentIntensityAnalyzer* to retrieve the polarity of each text. To obtain an overall score for each corpus, the mean is calculated across all scores for the respective class. The results from the Sentiment Analysis are visualized in Figure 4 while the exact scores can be found in Appendix 6.

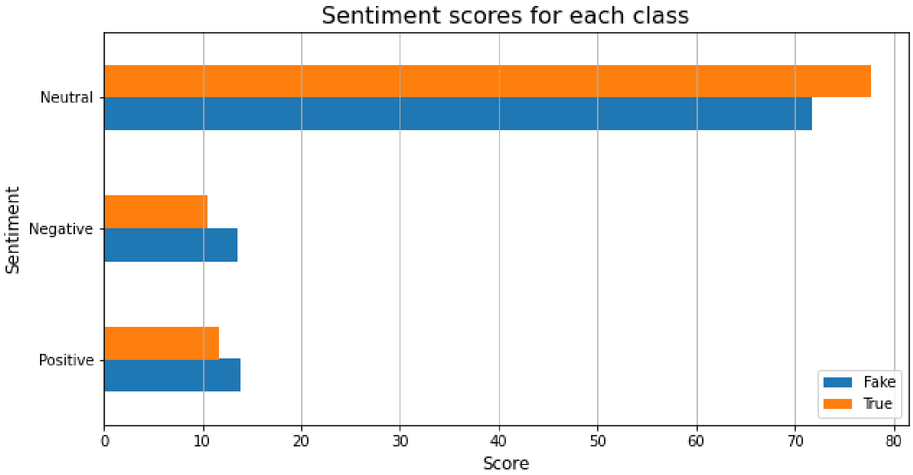


Figure 4 Average Sentiment scores per class

We can see that Fake News on average show a slightly higher tendency of being biased i.e. not neutral than True News. Additionally, if a True News text is considered as biased, there is a slightly higher likelihood of the text being positive rather than being negative. Nevertheless, these differences are rather small and would probably not allow for a clear distinction.

Altogether, all four analyses indicate some potential for a classification into True News and Fake News purely based on the textual structure and features. Still, if used alone, none of the techniques would probably be sufficient to classify each news instance accurately. Another implication that results from the analysis is that stopwords might be an informative feature too in this context. Therefore, it seems to be reasonable to train a classifier on a dataset that still contains stopwords and compare its performance to a model trained on data that had stopwords removed.

4.4 Classifier implementation

After analysing some features of the dataset in an isolated way, we finally want to examine how classification algorithms perform on the task of predicting the label for a given news article. For this, we decided to use Naïve Bayes as a rather simplistic model on the one hand and a rather complex NN approach which utilizes BERT embeddings on the other hand. A general description of each of the concepts can again be found in section 3.

Naïve Bayes

The Naïve Bayes framework was taken from the *NLTK* library. In order to apply the model and achieve the best possible results, the previously described pre-processing steps had to be performed first. Based on the results from the preceding section, the model was trained on two different inputs. One of them consisted of the articles with, the other one without stopwords. Furthermore, the words had to be encoded because the algorithm can only use numeric inputs. The same had to be done with the labels, where a 1 indicates a fake, a 0 indicates a truthful article there. In order to rank the articles, the 1500 most used words were selected as features of the model.

BERT

Classification via BERT was implemented using the *Tensorflow* library. Since BERT is to be regarded as a benchmark, the previously used pre-processing was applied. As usual in the application of BERT, stopwords were removed from the articles. An architectural overview of the model can be found in appendix 7.

Due to the high computing power of the model, UCloud was used as cloud computing platform. The selected resource has 5 vCPUS and 376 GB RAM. Furthermore, the model was optimized in 5 epochs.

4.5 Results

For the Naïve Bayes, a high accuracy can be observed for both variants. The fully pre-processed data that had stopwords removed achieves an accuracy of 97.95%. Surprisingly and yet in accordance with our findings from section 4.3, the model that is trained on the data that still includes stopwords reached an even higher value of 98.27% accuracy.

Nevertheless, the highest performance is achieved with BERT. The trained ANN with the BERT encoder achieves an accuracy of 99.988% and a loss of approximately 0.0782% (see Appendix 8&9).

Altogether, this suggests that a detection of Fake News purely based on their textual structure and features is fairly possible and works even better than anticipated from what we observed during the analysis phase.

5 Discussion

Since both applied models performed very well (both Naive Bayes approaches reached an accuracy of over 0.97 and the BERT approach reached an accuracy of over 0.99 (see section 4.5 for details) on the task of Fake News detection, we can answer our research question by concluding that textual data itself provides enough information to accurately predict whether an article stems from a Fake News outlet. By comparing a traditional classifier (NB) with a pretrained state of the art encoder for an ANN (BERT) we contribute to the overall discussion of Fake News detection.

Nevertheless, there are several limitations to the presented approach that could possibly distort the results of the classification or at least favour overfitting of the model. One first issue is related to the source of the news text. While a lot of the Fake News articles in our dataset seem to be scraped from Twitter, the portion is much smaller for the truthful articles. This can potentially lead to a classification bias, resulting in the misclassification of truthful articles scraped from Twitter as “Fake”. Future research should take this issue into consideration and try to reduce class imbalance and its potential side-effects.

Another flaw in the data is its limited variety with regard to the discussed topics. This again can potentially mitigate the generalizability of the classifier for news texts on varying, possibly unseen topics. Future work could use a more heterogeneous dataset in order to tackle this problem.

One final limitation of the approach refers to the robustness of the classifiers against fallacy. As the models can only predict a label based on the given news article and its textual structure without taking any further information into account, it might be more prone to being fooled by Fake News that adopt the writing structure of True News. Therefore, the more distributors of Fake News learn how to replicate the structure of True News, the stronger the need for integration of additional dimensions as inputs for the classification algorithm becomes.

6 Conclusion

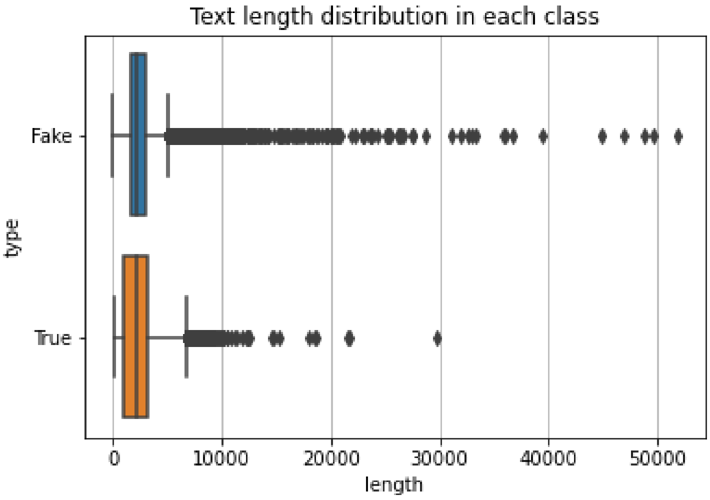
In this study, we focussed on the detection of Fake News based on textual data. We therefore first gave a structured overview of the existing literature within the field of Fake News detection. After that, we explained the most relevant techniques used in this paper, before outlining our concrete methodology. Such methodology included a Pre-processing phase as well as the application of two ML models: A traditional classifier (NB) and a pretrained state of the art encoder for an ANN (BERT). We then interpreted the results and compared both approaches, which enabled us to properly answer our initial research question. From the accuracy of our two models (over 97 % for both NB variants (see section 4.5 for details) and over 99% for BERT), we concluded that textual data itself provides enough information to accurately predict whether an article stems from a Fake News outlet. Finally, we outlined the limitations of our study and derived implications for future work. By comparing a traditional classifier with a pretrained state of the art encoder for an ANN and answering our initial research question, we contribute to the overall discussion of Fake News detection. Since the amount of inaccurate and manipulated information is constantly increasing, future work should build on the insides of this study to further develop and improve automatised Fake News detection systems.

References

- Ahmed, H., Traore, I., & Saad, S. (2018). Detecting opinion spams and fake news using text classification. Security and Privacy, 1(1), e9.
- Bovet, A., & Makse, H. A. (2019). Influence of fake news in Twitter during the 2016 US presidential election. Nature communications, 10(1), 1-14.
- Dagan, I., Dolan, B., Magnini, B., & Roth, D. (2010). Recognizing textual entailment: Rational, evaluation and approaches–erratum. Natural Language Engineering, 16(1), 105-105.
- Della Vedova, M. L., Tacchini, E., Moret, S., Ballarin, G., DiPierro, M., & de Alfaro, L. (2018, May). Automatic online fake news detection combining content and social signals. In 2018 22nd conference of open innovations association (FRUCT) (pp. 272-279). IEEE.
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
- Farokhian, M., Rafe, V., & Veisi, H. (2022). Fake news detection using parallel BERT deep neural networks. arXiv preprint arXiv:2204.04793.
- Feldman, R. (2013). Techniques and applications for sentiment analysis. Communications of the ACM, 56(4), 82-89.
- Giachanou, A., Zhang, G., & Rosso, P. (2020, October). Multimodal multi-image fake news detection. In 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA) (pp. 647-654). IEEE.
- Hutto, C., & Gilbert, E. (2014, May). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In Proceedings of the international AAAI conference on web and social media (Vol. 8, No. 1, pp. 216-225).
- Jurafsky, D., & Martin, J. H. (2022). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.
- Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.
- Kula, S., Choraś, M., & Kozik, R. (2019, May). Application of the bert-based architecture in fake news detection. In Computational Intelligence in Security for Information Systems Conference (pp. 239-249). Springer, Cham.
- Lazer, D. M., Baum, M. A., Benkler, Y., Berinsky, A. J., Greenhill, K. M., Menczer, F., ... & Zittrain, J. L. (2018). The science of fake news. Science, 359(6380), 1094-1096.
- Lemann, N.: Solving the Problem of Fake News. The New Yorker (2017). http://www. newyorker.com/news/news-desk/solving-the-problem-of-fake-news

- Long, Y. (2017). Fake news detection through multi-perspective speaker profiles. Association for Computational Linguistics.
- Madhusudhan, S., Mahurkar, S., & Nagarajan, S. K. (2020, September). Attributional analysis of Multi-Modal Fake News Detection Models (Grand Challenge). In 2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM) (pp. 451-455). IEEE.
- Oshikawa, R., Qian, J., & Wang, W. Y. (2018). A survey on natural language processing for fake news detection. arXiv preprint arXiv:1811.00770.
- Palić, N., Vladika, J., Čubelić, D., Lovrenčić, I., Buljan, M., & Šnajder, J. (2019). TakeLab at SemEval-2019 Task 4: Hyperpartisan News Detection. In Proceedings of the 13th International Workshop on Semantic Evaluation (pp. 995-998).
- Pandey, C. (2021). redBERT: A topic discovery and deep sentiment classification model on COVID-19 online discussions using BERT NLP model. International Journal of Open Source Software and Processes (IJOSSP), 12(3), 32-47.
- Potthast, M., Kiesel, J., Reinartz, K., Bevendorff, J., & Stein, B. (2017). A stylometric inquiry into hyperpartisan and fake news. arXiv preprint arXiv:1702.05638.
- Shu, K., Mahudeswaran, D., Wang, S., Lee, D., & Liu, H. (2018). Fakenewsnet: A data repository with news content, social context and spatialtemporal information for studying fake news on social media. arXiv preprint arXiv:1809.01286.
- Szczepański, M., Pawlicki, M., Kozik, R., & Choraś, M. (2021). New explainability method for BERT-based model in fake news detection. Scientific reports, 11(1), 1-13.
- Tacchini, E., Ballarin, G., Della Vedova, M. L., Moret, S., & de Alfaro, L. (2017). Some like it hoax: Automated fake news detection in social networks. arXiv preprint arXiv:1704.07506.
- Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(11).
- Vlachos, A., & Riedel, S. (2014, June). Fact checking: Task definition and dataset construction. In Proceedings of the ACL 2014 workshop on language technologies and computational social science (pp. 18-22).
- Wattenberg, M., Viégas, F., & Johnson, I. (2016). How to use t-SNE effectively. Distill, 1(10), e2.
- Zhang, L., Wang, S., & Liu, B. (2018). Deep learning for sentiment analysis: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4), e1253.

Appendix



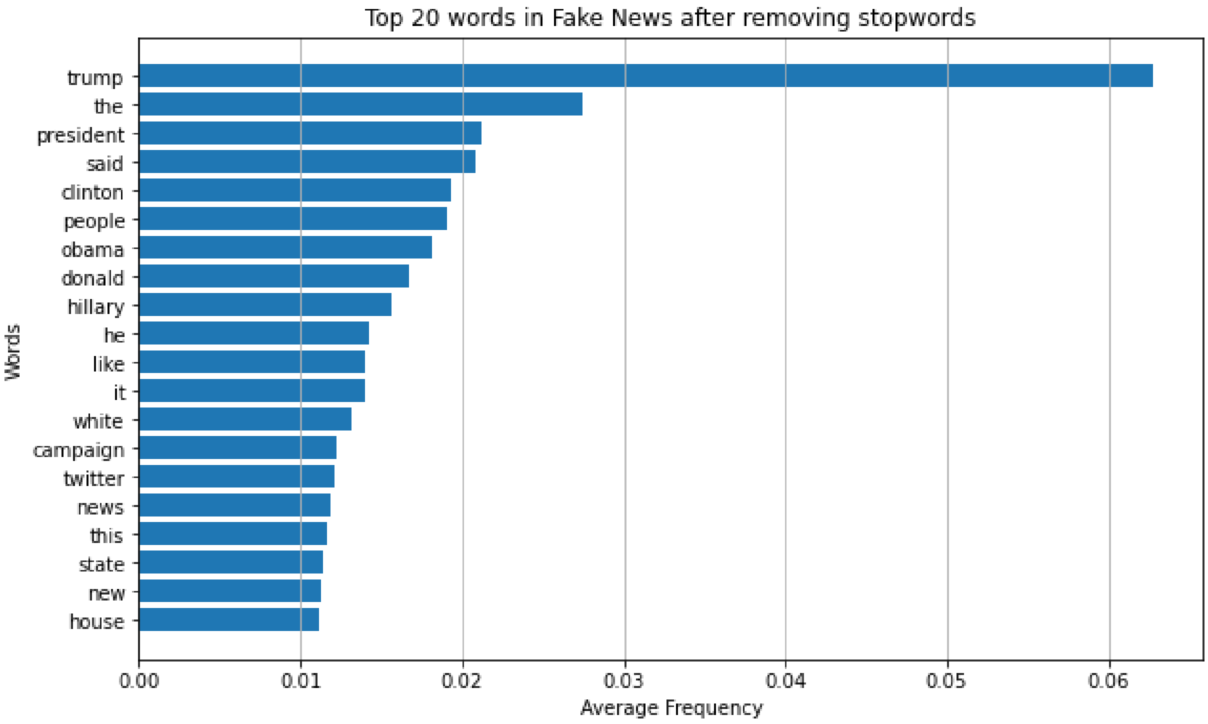
Appendix 1: Text length distribution per class

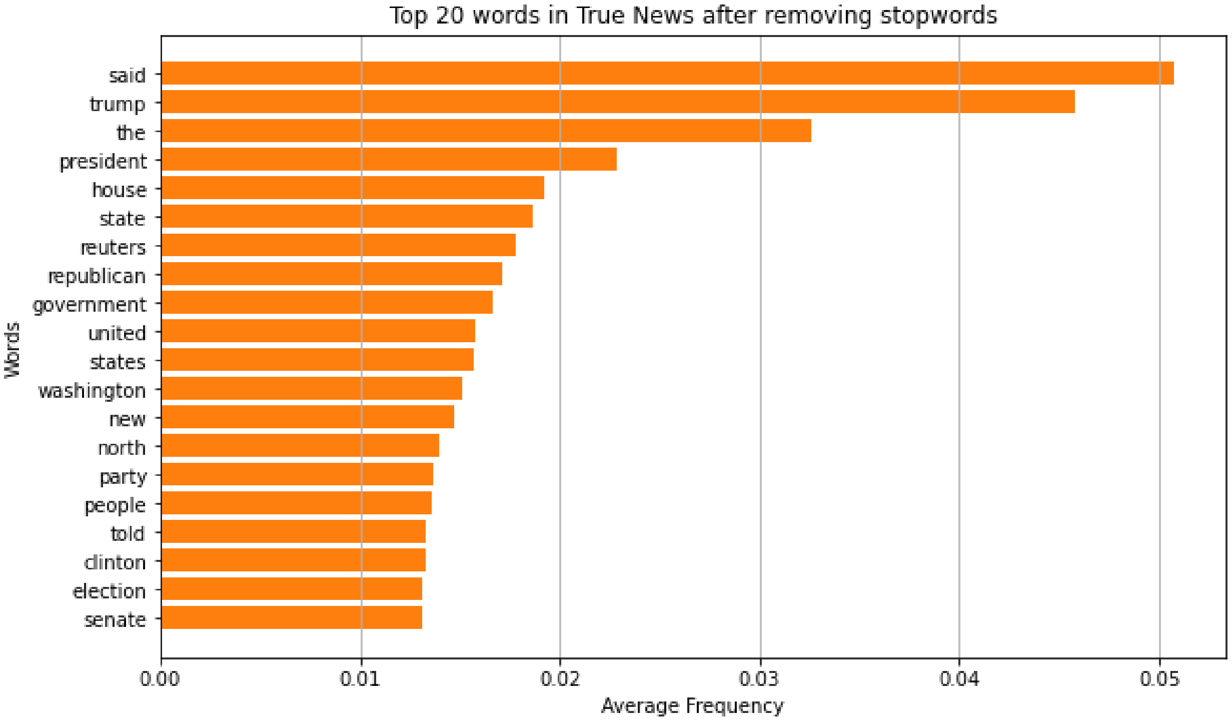
	Altered example sentence
Original	‘Donald Trump just couldn t wish all Americans a Happy New Year and leave it at that. Instead, he had to give a shout out to his enemies, haters and the very dishonest fake news media.’
Destopped*	‘Donald Trump couldn t wish Americans Happy New Year leave Instead shout enemies haters dishonest fake news media The reality star job couldn t As Country rapidly grows stronger smarter I want wish friends supporters enemies haters dishonest Fake News Media’
Sent-tokenized (with stopwords)	['Donald Trump just couldn t wish all Americans a Happy New Year and leave it at that.', 'Instead, he had to give a shout out to his enemies, haters and the very dishonest fake news media.']
Sent-tokenized (without stopwords)	['Donald Trump couldn t wish Americans Happy New Year leave', 'Instead shout enemies haters dishonest fake news media']
Word-tokenized (with stopwords)	['Donald', 'Trump', 'just', 'couldn', 't', 'wish', 'all', 'Americans', 'a', 'Happy', 'New', 'Year', 'and', 'leave', 'it', 'at', 'that', '.', 'Instead', ',', 'he', 'had', 'to', 'give', 'a', 'shout', 'out', 'to', 'his', 'enemies', ',', 'haters', 'and', 'the', 'very', 'dishonest', 'fake', 'news', 'media']
Word-tokenized (without stopwords)	['Donald', 'Trump', 'couldn', 't', 'wish', 'Americans', 'Happy', 'New', 'Year', 'leave', 'Instead', 'shout', 'enemies', 'haters', 'dishonest', 'fake', 'news', 'media',]
Lemmatized (with stopwords)**	‘Donald Trump just couldn t wish all Americans a Happy New Year and leave it at that . Instead , he had to give a shout out to his enemy , hater and the very dishonest fake news medium .’
Lemmatized (without stopwords)	'Donald Trump couldn t wish Americans Happy New Year leave Instead shout enemy hater dishonest fake news medium’

*Punctuation was stripped off in order not to miss out stopwords that directly follow or are followed by punctuation.

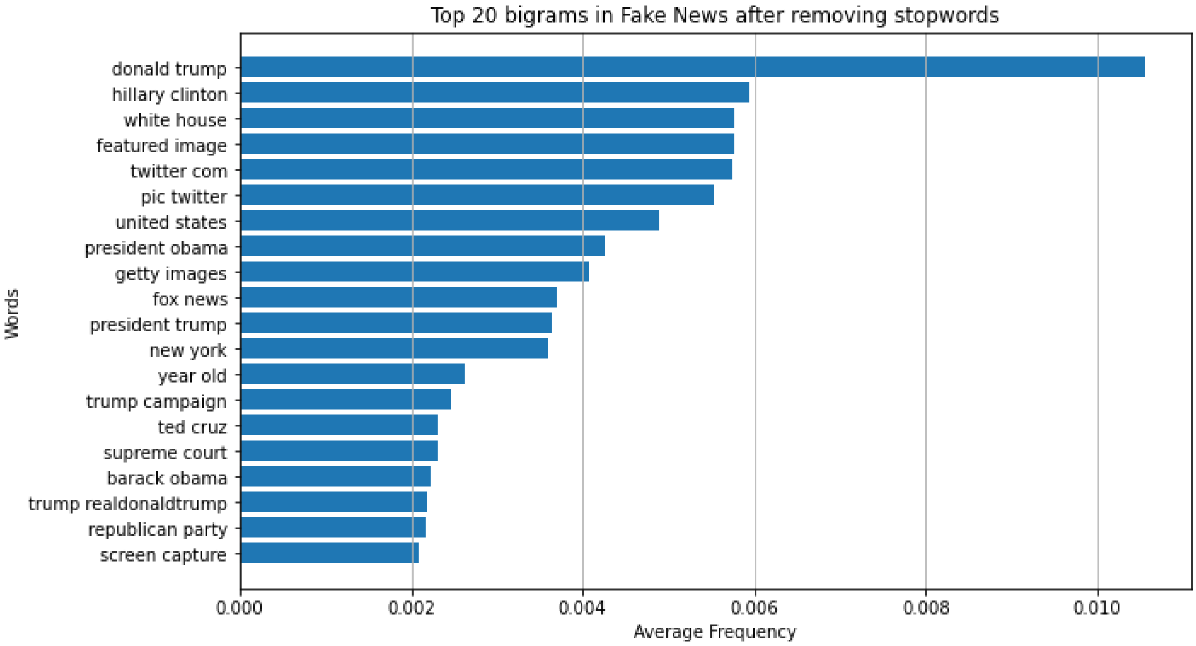
**Lemmatized data was concatenated after lemmatization.

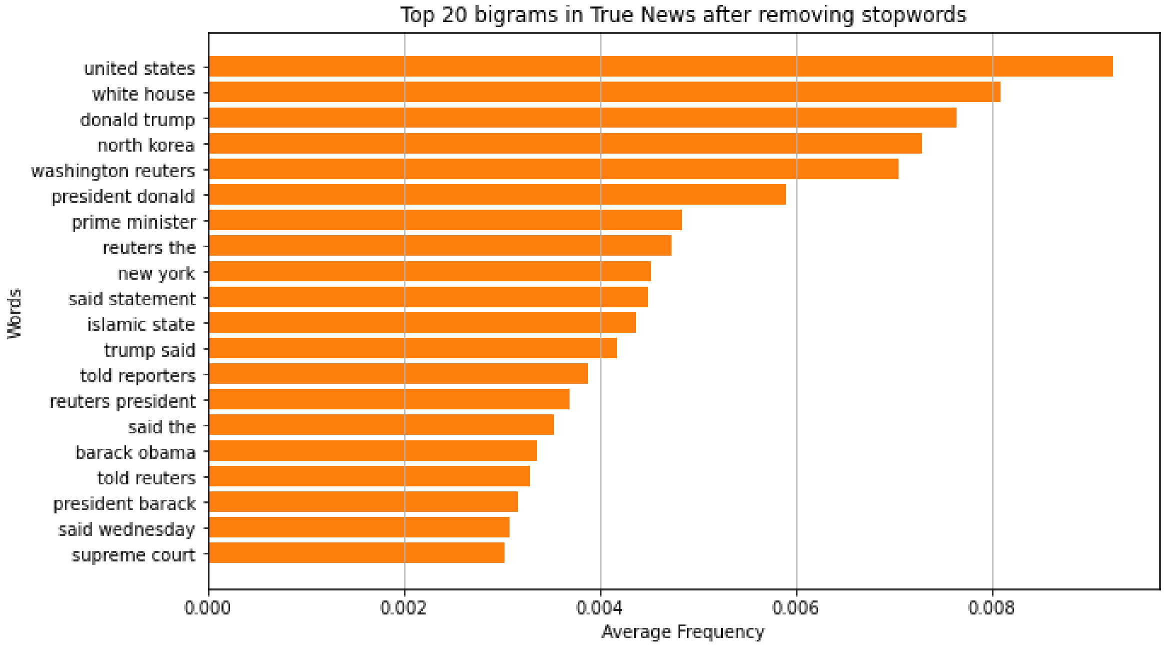
Appendix 2: Examplary pre-processing process



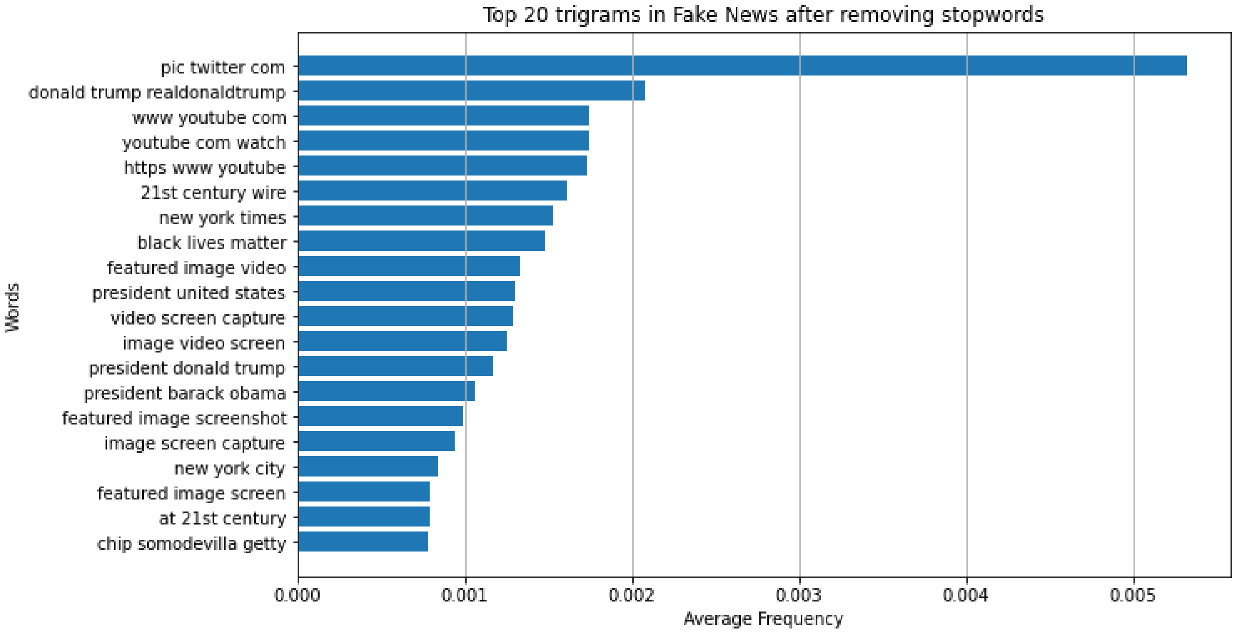


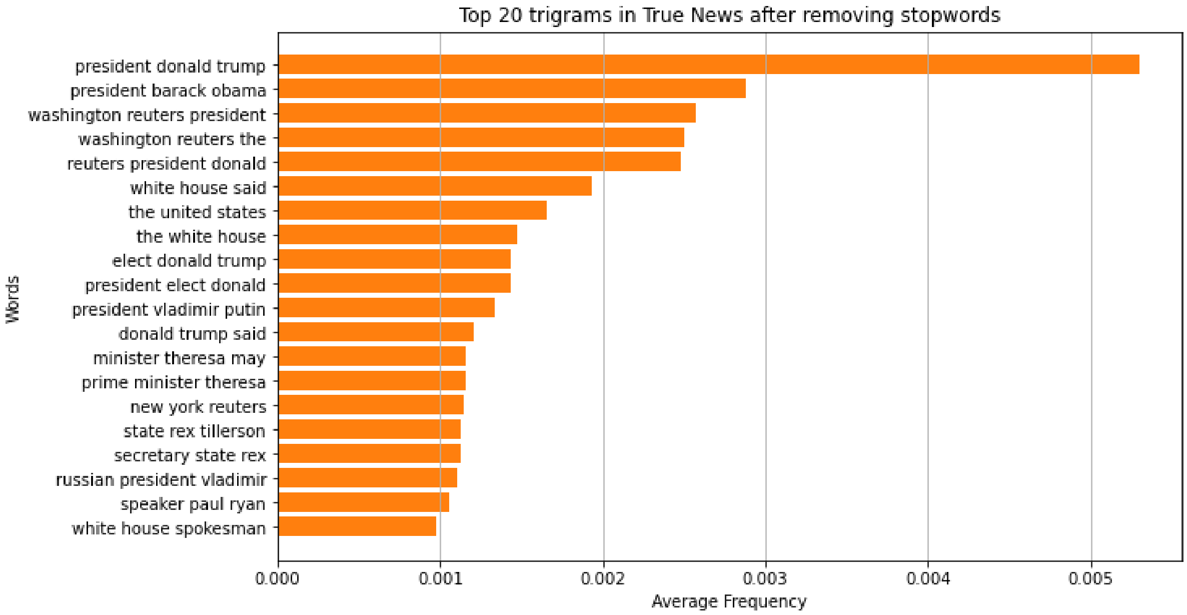
Appendix 3: Most frequent Unigrams





Appendix 4: Most frequent Bigrams

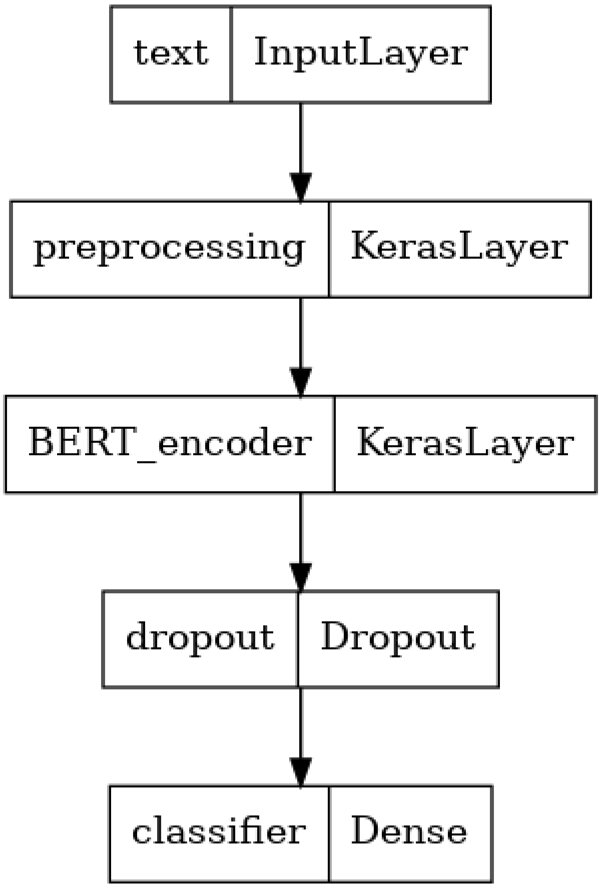




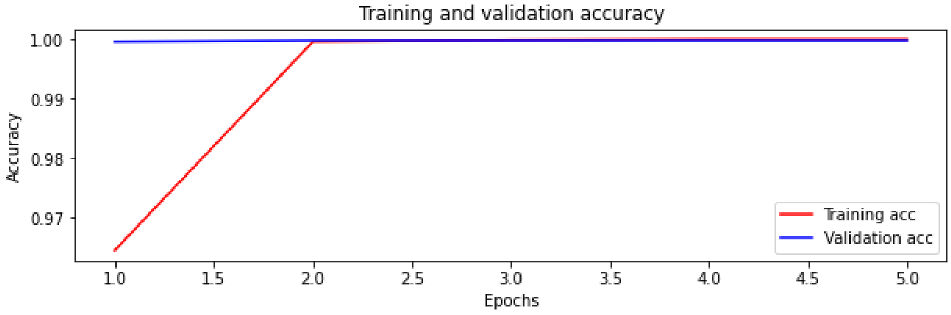
Appendix 5: Most frequent Trigrams

	Positive	Negative	Neutral
Fake News	13.797	13.510	71.773
True News	11.628	10.574	77.647

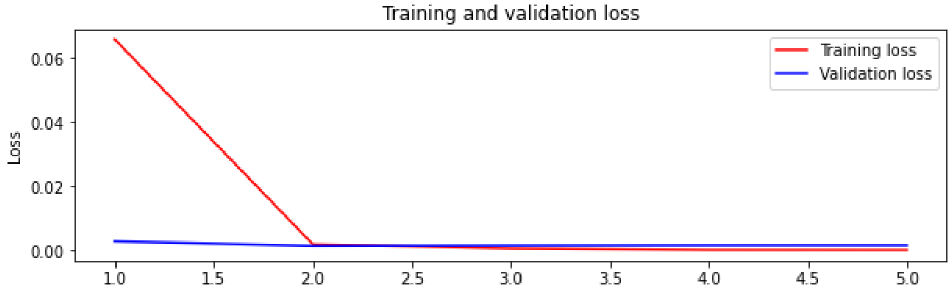
Appendix 6: Average Sentiment scores



Appendix 7: BERT Model – Architectural Overview



Appendix 8: BERT Model – Training and Validation Accuracy



Appendix 9: BERT Model – Training and Validation Loss