


Article

A Comparative Study of Machine Learning and Deep Learning Techniques for Fake News Detection

Jawaher Alghamdi ^{1,2,*}, Yuqing Lin ¹  and Suhuai Luo ¹

¹ School of Information and Physical Sciences, College of Engineering Science and Environment, The University of Newcastle, Newcastle 2308, Australia

² Department of Computer Science, King Khalid University, Abha 62521, Saudi Arabia

* Correspondence: jawaher.alghamdi@uon.edu.au

Abstract: Efforts have been dedicated by researchers in the field of natural language processing (NLP) to detecting and combating fake news using an assortment of machine learning (ML) and deep learning (DL) techniques. In this paper, a review of the existing studies is conducted to understand and curtail the dissemination of fake news. Specifically, we conducted a benchmark study using a wide range of (1) classical ML algorithms such as logistic regression (LR), support vector machines (SVM), decision tree (DT), naive Bayes (NB), random forest (RF), XGBoost (XGB) and an ensemble learning method of such algorithms, (2) advanced ML algorithms such as convolutional neural networks (CNNs), bidirectional long short-term memory (BiLSTM), bidirectional gated recurrent units (BiGRU), CNN-BiLSTM, CNN-BiGRU and a hybrid approach of such techniques and (3) DL transformer-based models such as BERT_{base} and RoBERTa_{base}. The experiments are carried out using different pretrained word embedding methods across four well-known real-world fake news datasets—LIAR, PolitiFact, GossipCop and COVID-19—to examine the performance of different techniques across various datasets. Furthermore, a comparison is made between context-independent embedding methods (e.g., GloVe) and the effectiveness of BERT_{base}-contextualised representations in detecting fake news. Compared with the state of the art's results across the used datasets, we achieve better results by solely relying on news text. We hope this study can provide useful insights for researchers working on fake news detection.

Keywords: fake news; misinformation; machine learning; deep learning; transformer-based models



Citation: Alghamdi, J.; Lin, Y.; Luo, S. A Comparative Study of Machine Learning and Deep Learning Techniques for Fake News Detection. *Information* **2022**, *13*, 576. <https://doi.org/10.3390/info13120576>

Academic Editor: Kostas Vergidis

Received: 27 October 2022

Accepted: 3 December 2022

Published: 12 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Traditionally, people fundamentally consume news and information through newspapers and television (TV) channels. However, with the advent of the Internet and its intrusion into our lifestyle, the former has become less prominent [1]. Today, social media and live streaming platforms play a fundamental role compared with television as major news sources, as 62 percent of U.S. people gained news from social media in 2016, while 49 percent watched news through social media in 2012 (<http://www.journalism.org/2016/05/26/news-use-across-social-media-platforms-2016/>) (accessed on 3 March 2022).

Nowadays, the role of online social networks (OSNs) has significantly increased due to their convenient access. As a result, it is no longer limited to being a window for communication between individuals; rather, it has become an important tool for exchanging information and influencing and shaping public opinions [2].

The other side of the coin is fake news dissemination, specifically on OSNs, which poses a great concern to the individual and society due to the lack of control, supervision and automatic fact-checking, leading to low-quality and fake content generation. As a result, the general public is prone to countless disinformation and misinformation on OSNs, including fake news (i.e., news stories created intentionally with falsified information to mislead readers) [3,4]. Figure 1 illustrates the increase in fake news cases over the last two years.



Figure 1. Fake news trends (2019–2022) [5].

It is not surprising to see falsehoods in information disseminated rapidly on OSNs, generating some degree of anonymity. For example, fake news has led to significant impacts on real-world events, where a piece of fake news from Reddit caused a real shooting (<https://www.rollingstone.com/politics/politics-news/anatomy-of-a-fake-news-scandal-125877/>) (accessed on 15 March 2022). In the 2016 U.S. presidential election, for instance, over one million posts were found to be related to a piece of fake news known as PIZZAGATE (<https://tinyurl.com/z38z5zh>) (accessed on 15 June 2022). Furthermore, during this period, the top 20 fake news pieces were reported to be larger than the top 20 most discussed real stories (<https://tinyurl.com/y8dckwhr>) (accessed on 15 June 2022). As stated by research on fake news velocity, tweets including falsified information on Twitter reach people six times faster than trustworthy tweets [6], resulting in fear, panic and financial loss in society [7]. According to a report in China, fake information constitutes more than one third of trending events on microblogs [8]. All these indicate how terribly fake news disseminates and how it can have an adverse social impact. According to [9], on Twitter, fake news, particularly political news, is usually retweeted by more users and disseminates extremely rapidly. In fact, some popular sources of information considered genuine, such as Wikipedia, are also subject to false information or fake news [10].

Without news verification, fake news would spread rapidly through OSNs, resulting in real consequences [11]. Therefore, researchers in the field of NLP dedicate their efforts to detecting fake news using various ML and DL algorithms. To better understand the performance of these algorithms and explore the directions for future study, in this paper, using four real-world fake news datasets, we compare the performance of seven classical ML algorithms—LR, SVM, NB, DT, RF, XGB and a voting ensemble ML method—with two scenarios of word representation methods: statistical (sparse) word vector representation methods and context-free (dense) pretrained word representation models. In addition, we also compare the performance of eight advanced ML models and two advanced transformer-based models: CNN, BiLSTM, BiGRU, CNN-BiLSTM, CNN-BiGRU, a hybrid model of CNN-BiLSTM-BiGRU (with two types of text representation methods: context-free and context-aware embedding models), $BERT_{base}$ and $RoBERTa_{base}$. Note that the presented approaches concentrate only on the structure without considering word sense (i.e., resolving ambiguities in context).

The key contributions of this paper are as follows:

- This paper surveys various feature-based methods and an assortment of ML and state-of-the-art transformer-based models used in the literature for fake news detection.
- This paper provides a benchmark study for a wide range of classical and advanced ML algorithms with pretrained word-embedding methods (i.e., context-free and context-aware) as well as advanced pretrained transformer-based models using various datasets.
- The experimental results indicate that no single technique can deliver the best performance scores across all datasets.
- In general, advanced PLMs such as $BERT_{base}$ and $RoBERTa_{base}$ are effective at detecting fake news.

2. Problem Definition

We consider the problem of fake news detection as a binary classification, where a model is created to predict the credibility of a piece of news as either fake or real based on

given attributes. Formally, assume $A = \{a_1, a_2, \dots, a_L\}$ is a set of news articles, statements or tweets (where L is the length of the input). The goal of fake news detection is to predict whether the input text represents fake or real content.

3. Related Works

ML and DL techniques have significantly contributed to detecting and categorising fake news. Earlier research in fake news detection mainly relied on linguistic features to engineer relevant information. The textually based features were divided broadly into two groups, according to [12]: (1) general features and (2) latent features.

3.1. Classical ML Algorithms

General textual features are those typically used with a traditional ML context. Such features are the catalyst for describing content style w.r.t. four linguistic language levels: lexicon, syntax, discourse and semantic [13]. At the lexicon level, some statistical techniques, such as the bag-of-words (BoW) model, are mainly applied to evaluate the frequency statistics of lexicons [14]. The main task of the syntax level is to assess part-of-speech (POS) (e.g., nouns and verbs) frequencies using part-of-speech (POS)-taggers [14,15]. At the discourse level, rhetorical parsing tools and rhetorical structure theory (RST) can be used to capture rhetorical relations as features based on their frequencies among sentences [14,16].

For example, Pisarevskaya [17] modelled language markers and rhetorical relations using SVM and random forest classifiers and achieved an F-score of 0.65. At the semantic level, these frequencies can be assigned to lexicons or expressions corresponding to each psycho-linguistic category [12], such as those determined in Linguistic Inquiry and Word Count (LIWC) [18]. The underlying assumption of text-based methods is that textually and linguistic-based features differ between fake and real claims.

It is common for fake news stories to contain inflammatory and opinionated language, since the purpose of their creation is usually financial or political gain, as opposed to reporting objective information. In addition, fake news content often contains clickbait (i.e., to incentivise consumers to read the full article by clicking on a provided link) or incites confusion [19]. Linguistic-based features (e.g., lexical and syntactic features) refer to those features that can capture particular writing styles and sensational headlines [20].

Different linguistic-based features were extracted from the news text content and explored by existing works for fake news detection. In [18], the authors used a set of linguistic-based features and conducted learning experiments to develop robust fake news detectors. First, they analysed the various features of the news articles (including n-grams, LIWC, punctuation, grammar and readability). Then, based on these features, a linear SVM classifier was trained. In this study, computational linguistics was shown to be useful in detecting fake news automatically. Fuller et al. [21] proposed a linguistic-based method for deception detection consisting of 31 linguistic features, where 3 classifiers were to be used to refine them to only 8 cues. Such clues were based on the different feature sets in the linguistic field proposed earlier [22,23], in addition to others.

The use of a relatively simple approach based on term frequency (TF) and term frequency-inverse document frequency (TF-IDF) has been shown to be effective in some previous studies. Riedel et al. [24] applied a multilayer perceptron (MLP) in the context of the fake news challenge dataset, and they showed that using a relatively simple approach based on TF and TF-IDF yielded a good accuracy score of 88.5%. Ahmed et al. [25] applied a linear support vector machine (LSVM) using TF-IDF and with unigram features on a dataset of 2000 news pieces, yielding an accuracy of 92%. Bharadwaj [26] experimented with different features such as TF and TF-IDF with n-gram features, and the results showed that random forest (RF) with bigram features achieved the best accuracy of 95.66%. In [27], the authors experimented with character and word n-grams to study their effect on detecting fake news and concluded that the former contributed more towards improving fake news detection performance compared with word n-grams. The TF-IDF method and CountVectorizer (CV) were used by [28] as a feature extraction technique.

They demonstrated that their approach was more accurate than state-of-the-art approaches such as Transformers' Bidirectional Encoder Representations (BERT). Linguistic features have been used in both supervised and unsupervised learning approaches. It is essential to recognise that prospective deceivers use short sentences, certain phrasal verbs as well as certain verb tenses as part of their language, which has been revealed by some cooperative experiments between psychologists, linguists and computer scientists [22,29–32]. For example, 16 language variables were investigated by Burgoon et al. [29] in order to see if they may help distinguish between deceptive and truthful communications. They conducted two experiments in order to construct a database, in which they set up face-to-face or computer-based discussions with one volunteer acting as the deceiver and the other acting genuinely. Then, such discussions were transcribed for further processing, and they came up with certain linguistic cue classes that could reveal the deceiver. The authors utilised the C4.5 decision tree (DT) technique with 15-fold cross-validation to cluster and construct a hierarchical tree structure of the proposed features. Their proposed method achieved an overall accuracy of 60.72 per cent on a short sample of 72 cases. They concluded that noncontent words (e.g., function words) should be included when studying social and personality processes. According to the authors, in addition to specific nouns and verbs, linguistic style markers such as pronouns, articles and prepositions are equally important in revealing what someone is thinking and feeling. Therefore, liars are more likely to tell less self-relevant, less complicated and more negative stories. Moreover, liars were more likely than truth-tellers to utilise negative emotion (more negative feeling) terms [31].

Liars may feel guilty, either for the lie they said or for the topic they lied about [33]. Knapp et al. [34] observed that liars were far more likely than truth-tellers to make disparaging statements about their communication partners. In addition, they observed that the use of other references is more common among liars than among truth-tellers. However, this is inconsistent with what [31] found, where they observed that compared with truth-tellers, liars used fewer third-person pronouns. Horne et al. [35] applied an SVM classifier with a linear kernel using several linguistic clues. The authors cast the problem as a multi-class classification to identify whether an article was real, fake or satire, where classes were equally distributed. After a 5-fold cross-validation with BuzzFeed News (<https://www.buzzfeednews.com/article/craigsilverman/viral-fake-election-news-outperformed-real-news-on-facebook>) (accessed on 5 May 2022) that was enriched with satire articles, they reached 78 percent accuracy. Their feature set mainly consisted of POS tags and specific LIWC word categories.

A wide range of stylistic, morphological and grammatical text and punctuation could serve as useful cues to detect news veracity. These sets of features were adopted by Papadopoulou et al. [36] using a two-level text-based classifier in order to detect click-bait. Similarly, Rubin et al. [37] used some of these features, such as punctuation and grammatical features. Sentiment analysis, opinion mining and opinionated and inflammatory language as factors for fake news detection have also been explored [19,37–46]. Bhelande et al. [47] applied a naive Bayesian classifier using a bag of positive and negative words (i.e., sentiment analysis). Syntactic features (e.g., sentence-level features such as POS and n-grams) [4,13,18,25,37,39,44,48–50], and lexical features for salient content words (i.e., character-level and word-level features) [8,18,20,25,35,37,40,44,51–56] have been adopted by existing research. Castillo et al. [40] analysed and assessed information credibility on Twitter by adopting a list of rudimentary content-based features. In [57], the authors aimed to gauge the credibility of web claims by exploiting language stylistic features. They claimed that the language style of the reporting articles plays a crucial role in understanding their credibility. Da Silva et al. [58] examined ML approaches to identify and detect fake news. They found that neural networks composed of statistical classification algorithms heavily concentrate on analysing lexicons as the main features for detection to perform best. Potthast et al. [20] used news content to extract style features for fake news prediction. Psycholinguistic cues (i.e., additional indicators of persuasive language) such as sadness, anger and others and signals of biased language have been adopted in

detection as well [9,18,37,46,49,54]. More work on fake news detection relying on linguistic cues can be found in [52,59,60]. A systematic mapping study was conducted in [61] to analyse and synthesise studies regarding the use of ML techniques for detecting fake news. Klyuev [62] discussed different approaches and techniques to fight fake news. They also discussed how important it is to define text features using NLP methods to build a text document profile. However, it turns out that straightforward approaches such as n-grams and POS tags tend to ignore different writing style features and cannot capture cues across long news articles. Indeed, irrelevant (noisy) text inevitably exists in fake news datasets, particularly those extracted from OSNs such as Twitter. If not processed, this would end up with classification performance leading to inaccurate predictions. This poses the challenge of how to encode such data so as to mitigate such a problem. Various dedicated efforts have been put forward to respond to this challenge using neural network-based models. Each provides a unique set of informative and useful features that can help discriminate fake news from real news.

3.2. Advanced ML and DL Models

Latent textual features are those used for news text representations which can be applied at the word level [63,64], sentence level [65,66] or document level [66], resulting in vector representations that can be used for further processing (i.e., as an input to a classifier). For example, one study on detecting rumours on Twitter applied Word2Vec in order to create vector representation [67,68]. Another study on detecting fake news based on the content [69] applied convolutional neural networks (CNNs) and bidirectional long short-term memory (BiLSTM) for embedding the textual and speaker metadata information for fake news detection. Quian et al. proposed two-level CNNs, where the first level produces embedding for a sentence using words and the second level uses the sentence embedding to generate article embedding [70]. Using CNNs and pretrained word embeddings, Goldani et al. [71] proposed a capsule network model based on the ISOT and LIAR datasets for (binary and multi-class) fake news classification. Their results showed that the best accuracy obtained using binary classification on ISOT was 99.8%, while multi-class classification using the LIAR dataset yielded 39.5% accuracy. Similarly, Girgis et al. [72] performed fake news classification using the LIAR dataset. The authors employed three different models: a vanilla recurrent neural network (RNN), a gated recurrent unit (GRU) and long short-term memory (LSTM). Regarding accuracy, the GRU model resulted in 21.7% accuracy, slightly higher than the other two models (LSTM = 21.6% and RNN = 21.5%). The authors presented a DL model in [73] for automatically detecting fake news in Slovak. In addition, several local online news sources were used to gather data related to the COVID-19 pandemic to train and evaluate various DL models. A model combining a bidirectional long-short-term memory network with one-dimensional convolutional layers achieved an average macro F1 score of 94% on an independent test set. Thus, advanced ML methods perform substantially better when used to classify fake news on a binary scale but perform much worse when used to classify fake news on a more refined level.

In fake news detection, it is important to capture contextualised features. In contrast to those unidirectional models that read the text input sequentially, BERT offers clear advantages over encoding models in the literature, where it basically reads the entire sequence of words at once and flows it through a stack of transformer encoders. The transformer is simply an attention-based neural network model which alleviates the currency-related issues in RNNs by handling long-term dependency in the text more efficiently using an attention mechanism. The attention mechanism is a module that interprets how significant a word is for the subsequent representation of the current word by assigning each word a weight based on the context (relevance of all other words in different positions in a sentence against the current word at the current position). Generally speaking, unlike feature-based engineering methods, which failed to fully exploit the rich semantic and syntactic information in the content, the results obtained by DL-based methods show superior improvements. Furthermore, as a state-of-the-art model, BERT can capture deep contextualised informa-

tion. The technique of learning how to transfer knowledge is known as transfer learning, which stores and applies the knowledge gained from performing a specific task to another problem. Learning this way is useful for training and evaluating models with relatively small amounts of data. In recent years, pretrained language models (PLMs) have become mainstream for downstream text classification [74] thanks to transformer-based structures. Major advances have been driven by the use of PLMs, such as ELMo [75], GPT [76] or BERT [74]. BERT and RoBERTa, the most commonly utilised PLMs, were trained on huge corpora, such as those containing over three billion words for BERT [74]. The success of such approaches raises the question of how such models can be used for downstream text classification tasks. Over the PLMs, task-specific layers are added for each downstream task, and then the new model is trained with only those layers from scratch [74,77,78] in a supervised manner. Specifically, these models use a two-step learning approach. First, in a self-supervised manner, they learn language representations by analysing a tremendous amount of text. This process is commonly called pretraining. Second, feature-based and fine-tuning approaches can then be used to apply these pretrained language representations to downstream NLP tasks. The former uses pretrained representations and includes them as additional features for learning a given task. The latter introduces minimal task-specific parameters, and all pretrained parameters are fine-tuned for the downstream tasks. These models are advantageous in that they can learn deep context-aware word representations from large unannotated text corpora, namely large-scale self-supervised pretraining. This is especially useful when learning a domain-specific language with insufficient available labelled data.

As we will further discuss in later sections, this paper exploits the power of BERT_{base} and its variations in building robust fake news predictive models. Few studies have been conducted using such models, despite using different methodologies and different scenarios, which have shown promising results. One recent example is a study conducted by Kula et al. [79], which presents a hybrid architecture based on a combination of BERT and an RNN. Alghamdi et al. [80] presented a computational framework for automatic fake news detection using BERT_{base}. The authors applied BERT_{base} to encode the input text, and then the resulting output sequence was fed into a CNN network to extract the salient local features. The metadata were then encoded using a CNN followed by a BiLSTM and then a classification layer. Their study achieved state-of-the-art performance on the LIAR dataset. Aggarwal et al. [81] showed that BERT, even with minimal text preprocessing, provided better performance compared with that of the LSTM and gradient-boosted tree models. Jwa et al. [82] adopted BERT for fake news detection by analysing the relationship between the headline and the body text of news using the FNC dataset, where they achieved an F1 score of 0.746. In an attempt to automatically detect fake news spreaders, Baruah et al. [83] proposed BERT for the classification task, achieving an accuracy of 0.690.

Given that BERT is more complex to train (depending on how large the number of parameters being used is), a variation of BERT, the so-called DistilBERT method [84], provides a simpler and more reasonable number of parameters compared with that of BERT (reducing BERT by 40% in size while retaining 97% of its language understanding abilities), thus leading to faster training (60% faster). With a larger dataset, larger batches and more iterations, a robust BERT model was developed: the so-called RoBERTa [78]. A benchmark study of ML models for fake news detection was provided in [85], where the authors formulated the problem of fake news detection using three different datasets, including the LIAR dataset [69] (see Section 4.6.1 for more details about the dataset), as a binary classification. Their experimental results showed the power of advanced PLMs such as BERT and RoBERTa. A summary of the previous related studies on the LIAR dataset (binary classification) can be seen in Table 1.

Table 1. A summary of exiting work on the LIAR dataset.

Models	Metrics			
	Accuracy	Precision	Recall	F1
RoBERTa _{base} [85]	0.62	0.63	0.62	0.62
SVM [86]	0.62	NA	NA	NA

Using a composite fake news dataset called FakeNewsNet (see Section 4.6.2 for more details about the dataset), which includes two datasets—PolitiFact and GossipCop—the authors of [87] employed an autoencoder with LSTM of two layers for both encoders and decoders and also used another network of LSTM with two layers to capture the temporal pattern of user engagements. The authors later presented a DL model based on the hierarchical attention network for fake news detection using the same datasets where their proposed framework achieved state-of-the-art results [88]. Using the PolitiFact dataset, a deep neural network model with various representations was proposed in [89]. A summary of the previous related work on the FakeNewsNet dataset can be seen in Tables 2 and 3.

Table 2. A summary of exiting work on (FakeNewsNet) PolitiFact dataset.

Models	Metrics			
	Accuracy	Precision	Recall	F1
Social Article Fusion [87]	0.69	0.64	0.79	0.71
Logistic Regression (N-Gram) [90]	0.80	0.79	0.78	0.78
BiLSTM-BERT [91]	0.8558	NA	NA	NA
LNN-KG [89]	0.880	0.9011	0.880	0.8892
DEFEND [88]	0.904	0.902	0.956	0.928

Table 3. A summary of exiting work on (FakeNewsNet) GossipCop dataset.

Models	Metrics			
	Accuracy	Precision	Recall	F1
CNN [87]	0.723	0.751	0.701	0.725
Logistic Regression (N-Gram) [90]	0.82	0.75	0.79	0.77
DEFEND [88]	0.808	0.729	0.782	0.755

In [92], the authors applied a pretrained transformer model, the so-called XLNet model, combined with Latent Dirichlet Allocation (LDA) by integrating contextualised representations generated from the former with topical distributions produced by the latter. Their model achieved an F1 score of 0.967. In the same vein, a fine-tuned transformer-based ensemble model was proposed in [93]. The proposed model achieved a 0.979 F1 score on the Constraint@AAAI2021-COVID19 fake news dataset (see Section 4.6.3 for more details about the dataset). Similarly, the authors of [94] carried out several experiments on the same dataset, and they proposed a framework for detecting fake news using the BERT language model by considering content information and prior knowledge as well as the credibility of the source. According to the results, the highest F1 scores obtained ranged from 97.57 to 98.13. By applying several supervised advanced ML models such as CNNs, LSTM and BERT to detect COVID-19 fake news, the authors of [95] achieved the best accuracy of 98.41% using the BERT-based version. A summary of the previous related studies on the COVID-19 dataset can be seen in Table 4.

Table 4. A summary of exiting work on the COVID-19 dataset.

Models	Metrics			
	Accuracy	Precision	Recall	F1
SVM+LR+NB+biLSTM [96]	NA	NA	NA	0.94
SVM [97]	0.9570	0.9571	0.9570	0.9570
SNN(LM+KG) [89]	0.9570	0.9533	0.9652	0.9569
BERT-Based [95]	0.9841	NA	NA	NA
Ensemble Transformer Models [93]	0.9799	0.9799	0.9799	0.9799
XLNet with Topic Distributions [92]	NA	0.968	0.967	0.967
SVM [98]	0.9332	0.9333	0.9332	0.9332

To summarise, classical ML algorithms are easy to comprehend and perform well on small datasets, but they (1) require complex feature engineering and (2) fail to capture substantial semantical contextual knowledge for a specific input text. To overcome this, advanced ML techniques such as CNN- and RNN-based methods are well suited for complicated classification problems powered by massive data and can learn more complicated (latent) features. However, CNNs typically struggle with capturing long-term contextual dependencies, while RNN-based methods perform suboptimally in handling such dependencies. As such, a combination of these two architectures may overcome some of their inherent limitations. Aside from the fact that surface-level features cannot effectively capture semantical patterns in text, the lack of sufficient data constitutes a bottleneck for DL models. Thus, to address this, the power of the DL transformer-based models, such as BERT and its variations, can be effectively leveraged to build robust fake news predictive models.

4. Comparative Study

Based on thorough literature review analysis, we conducted this comprehensive comparative study by selecting several popular, well-discussed classical and advanced ML and DL models for fake news detection. In this section, we will first present different word-embedding techniques developed in NLP and then introduce different traditional and advanced ML algorithms as well as DL models for fake news detection and the benchmark datasets, which will be used to conduct the comparative study. The objective is to examine the performance of a wide range of ML techniques, including classical and advanced ML models as well as DL approaches, across various datasets for fake news detection.

4.1. Embeddings

Word embedding is considered the centre of any NLP task and is basically a form of word representation that bridges the human understanding of language to that of a machine. Word embeddings are typically learned from a large corpus of text. The distributed representational vector that captures the semantics of the input text can be obtained using two main alternatives that have been used for several text classification tasks in NLP. Modelling text is challenging because it is messy, and techniques such as ML algorithms require well-defined, fixed-length inputs and outputs. ML algorithms require the text to be converted into numbers (specifically vectors of numbers) as they cannot directly work with raw text. These vectors capture more linguistic properties of the text, as “in language processing, the vectors x are derived from textual data, in order to reflect various linguistic properties of the text” [99] (p. 65). Textual data requires a numerical representation and, most importantly, an efficient representation for computation and manipulation. Various statistically and contextually based methods have been developed to represent text numerically. The former is based only on statistical metrics, which typically generate a sparse vector space, while the latter is based on the concept of word context, which produces a rather dense vector space.

Several approaches have applied such methods (e.g., the bag-of-words [100], n-gram [101] and TF-IDF methods [102]) as input for text classification using ML (e.g., NB classifiers [103], K-NN algorithms [104] and SVM [105]). Nevertheless, the contextualised

representation that allows for efficient computation and captures the underlying patterns is urgently needed, especially with the massive amounts of textual data. These statistically based representation methods are computationally efficient and have achieved promising results where they are traditionally considered the centre of any text classification task. However, these methods focus entirely on capturing the frequency features of a particular word, and the contextual information of a text is fully disregarded, making it difficult to capture semantic-based information. To capture more semantical information, pretrained language representation models (a.k.a context-independent models) were developed, such as Word2Vec and GloVe, which captured those semantics patterns but did little to capture context information. A significant amount of attention has been devoted to developing context-aware representations of textual data, including transformer-based models such as BERT, which has led to outstanding results in most NLP mainstream tasks. Overall, these techniques try to model the following problem. Assume we have a corpus of n documents $S = \{d_1, d_2, \dots, d_n\}$, each of which consists of m words $W = \{w_1, w_2, \dots, w_m\}$ and a vocabulary V . The embedding vector representation \vec{w}_i is defined as mapping each word w_i in a specific document d_i into a continuous space \mathbb{R}^d , where d is the dimension of the vector space. Mathematically, the words in a document can be mapped as follows:

$$w_i \rightarrow \vec{w}_i, \vec{w}_i \in \mathbb{R}^d \quad (1)$$

4.1.1. Non-Contextualised Embeddings: Sparse Vector Representation-Based

In this subsection, we present two statistical methods that generate sparse vector representations of documents. The first one is a popular and simple feature extraction method with text data: the bag-of-words (BoW) method. The second one is the TF-IDF method, which overcomes the problem of the former.

Bag of words (BoW): The bag-of-words model, or BoW for short, is very popular, simple and flexible, and it can be used in a myriad of ways for extracting features from the text in order to be used for modelling using ML algorithms. It is a representation of text that describes the occurrence of words within a specific document. The idea of this distributional representation of features was investigated and proposed by Harris [106]. The BoW model, as the name implies, discards any information about the order or structure of words in the document and is only concerned with whether known words occur in the document and not where they occur in the document, thus failing to capture semantic patterns. To illustrate this, given two sentences (the NLP is difficult, not easy at all, and the NLP is easy, not difficult at all) with completely opposite semantic meanings, the BoW model would give them the exact same representation just because they have the exact same words but in a different order, which is not effective. Another issue of such a model is that as the vocabulary size increases, so does the dimension of the vector space since, in this model, the number of words in the vocabulary forms the dimension of the vector representation of a document, resulting in what are called sparse representations. Such large dimensions, however, are bound to result in the so-called curse of dimensionality, making it easy to fall into overfitting and resulting in terrible out-of-sample performance.

Term frequency-inverse document frequency (TF-IDF): The problem with the BoW method is that it treats all words as equally important, and this is attributed to the fact that BoW methods score each word based on its frequency in a document. Thus, frequently occurring words dominate others in the document with a larger score, which is problematic, especially when such words are not as informative to the model as less-occurring words. Rescaling the frequency of words by penalising the scores of the most frequent words across all documents is one approach to dealing with this issue. This approach is the so-called TF-IDF metric proposed in [107], and it has been widely used in many NLP tasks. The TF-IDF method allows for quantifying words by reflecting how important a word is to a document in a corpus of documents. This method is premised on the idea that each word is assigned its own weight w_{ij} based on its appearance in the document and across all of the documents. These weights highlight words that are distinct and contain useful

information in a given document. “Thus the idf of a rare term is high, whereas the idf of a frequent term is likely to be low” [108] (p. 118). For each word in a document j , the TF-IDF value is calculated by first calculating the term frequency (TF), which counts the number of occurrences of words in a document, and then the inverse document frequency (IDF), which is the catalyst for ensuring that words appear less commonly are assigned more weights compared with those occurring more frequently (e.g., stop words), which is calculated as follows:

$$\log\left(\frac{|D|}{df_i}\right) \quad (2)$$

where df_i denotes the number of documents that contains a word i and $|D|$ refers to the number of documents in the corpus. The TF-IDF metric is calculated as follows:

$$w_i = tf_{ij} \cdot \log\left(\frac{|D|}{df_i}\right) \quad (3)$$

where tf_{ij} , df_i and $|D|$ refer to the number of appearances of a word i in the document j , the number of documents containing a word i and the number of documents, respectively. However, this method cannot capture semantic patterns, making it only useful for lexical features.

4.1.2. Non-Contextualized Embeddings: Dense Vector Representation-Based

Word embedding: Perhaps one of the key breakthroughs for the remarkable performance of ML methods in a suit of NLP tasks is a way of representing words (i.e., a learned representation) in a given text by allowing words with similar meanings to have a similar representation. It is this approach that generates a dense vector that carries more informative information. This can have many advantages, as “one of the benefits of using dense and low-dimensional vectors is computational: the majority of neural network toolkits do not play well with very high-dimensional, sparse vectors” [99] (p. 92). This also allows the model to generalise well. Contrary to classical word representation methods such as BoW, which generate sparse word representations using thousands or millions of dimensions, the rationale behind the word-embedding approach is premised on the idea of assigning each word a densely distributed representation (i.e., a real-valued vector) with often tens or hundreds of dimensions.

Word2Vec: Contrary to the previously discussed word representation methods that fail to capture the semantic and syntactic relations between the words, producing sparse vector representations, Word2Vec and GloVe (discussed next) can generate dense semantic representations. Word2Vec, developed by Tomas Mikolov et al. at Google in 2013 [109], is a statistical method that leverages the use of neural networks for efficiently learning a standalone word embedding from a given text corpus. This approach is considered a de facto standard for developing pretrained word embedding. The learned vectors of such an approach can be analysed, and interesting results can be found. Two different learning approaches have been proposed to model the algorithm architecture: the continuous bag-of-words (CBOW) and the skip-gram models. The CBOW model uses the context of a current word in order to predict that word. In other words, it learns the embedding by predicting the current word based on its context. Alternatively, the continuous skip-gram model learns by predicting the surrounding words given a current word. The Word2Vec approach is advantageous in that efficient and high-quality word embeddings can thankfully be learned with less space and time complexity, and this shows the key benefit of this approach, where it can handle larger corpora of a text by allowing larger dimensional embeddings to be learned (more dimensions) from such corpora. Here, the Gensim library is used to retrieve Word2Vec embeddings. The text is first cleaned, then stemming is used, and stop words are removed. The models were initialised with Word2Vec embeddings of a vector size set to 300-d and maximum length tokens set to 128.

GloVe: By extending the previous word-embedding approach (Word2Vec), the Global Vectors for Word Representation method, or GloVe for short, was developed by Pennington et al. [64] in order to learn word vectors more efficiently. This approach generally results in better word embeddings. This is because GloVe combines global statistics from matrix factorisation techniques such as latent semantic analysis (LSA) and local context-based learning methods such as Word2Vec. In short, “GloVe, is a new global log-bilinear regression model for the unsupervised learning of word representations that outperforms other models on word analogy, word similarity, and named entity recognition tasks.” [64]. Here, the models are initialised with 100-dimensional pretrained GloVe embeddings.

4.1.3. Contextualized Embeddings: Context-Aware Embeddings

Bi-Directional Encoder Representations from Transformers (BERT): PLMs (vector representations of words and embeddings) trained on massive amounts of textual data have formed the basis of many language-based tasks nowadays. As context-independent neural embeddings, Word2Vec and GloVe extracted from shallow neural networks are examples of the most frequently pretrained word embedding techniques prior to the advent of recent trends (PLMs shined in 2018). Yet, nevertheless, these techniques failed to capture deeper contextual relations, since they mostly model indirect relationships by capturing only short-range context based on a specific co-occurrence window. In fact, since 2018, the interest of the NLP community in these kinds of pretrained word-embedding techniques has constantly been fading in favour of the most recent trend of transfer learning. Examples include Universal Language Model Fine-Tuning (ULMFiT) [110], Embedding from Language Models (ELMo) [75], OpenAI Generative Pretrained Transformer (GPT) [76], and Google’s BERT model [74]. ULMFiT [110] is pretrained on a universal language model on a general domain corpus which then can be fine-tuned on the target task data. Radford et al. [76] generated a transformer-based language model, the so-called OpenAI GPT model, which is a unidirectional language model. Unlike OpenAI GPT, BERT, generated by Devlin et al. [74], is the first deeply bidirectional and unsupervised language representation that employs a multi-layer bidirectional transformer encoder which jointly conditions both the left and the right contexts in all layers. The transformer architecture comprises two blocks, an encoder and a decoder, to read the text and produce a prediction, respectively. BERT uses only the encoder portion of the transformer. BERT [74] is a language representation model which Google AI introduced. Before discussing how BERT works, we first discuss some key points behind its prominent success. As the first-of-its-kind language representation, BERT contains several transformer encoders stacked together that can be used to pretrain deep bidirectional representations. The concept of bidirectionality in BERT allows it to consider left and right contexts. In other words, BERT is based on a self-attention layer jointly conditioning both the left and the right contexts in all layers, and thus, BERT generates context-aware embeddings. This is the key differentiator between BERT and its predecessor OpenAI GPT, where the former is deeply bidirectional whereas the latter is a unidirectional pretrained model (left-to-right language model pretraining). More details about its architecture will be introduced later.

4.2. ML Algorithms

In this subsection, we mainly describe the classification models we used.

4.2.1. Classical ML Models

The six models that were chosen for investigation, as previously stated in the introduction, were SVM, LR, MNB, DT, RF and XGB with both TF-IDF features and pretrained Word2Vec representations. A plethora of ML algorithms have been explored and tested in the literature for fake news detection. Below is the list of those approaches implemented in this work:

- **Logistic Regression (LR):** LR is a statistical model applied as a great baseline algorithm in a wide range of text classification tasks.

- **Support Vector Machine (SVM):** The SVM classifier is a strong classifier that yields promising results in a suite of NLP tasks.
- **Multinomial Naive Bayes (MNB):** MNB is a kind of probabilistic algorithm (a Bayesian learning approach) that is also popular and yields great results in different NLP tasks.
- **Decision Tree (DT):** This is a tree-based algorithm whose end nodes represent high-level features. A branch represents an output, while a leaf represents a label class. There are internal nodes that test one attribute and branch from a node that selects one value for the attribute. The leaf node is used to predict the class label. Classification is carried out based on supervised learning, which involves mapping the features and values to desired outcomes.
- **Random Forest (RF):** RF models consist of a set of decision trees, each trained from a random selection of features.
- **XGBoost (XGB):** This is an ensemble ML algorithm. The XGB algorithm uses a gradient-boosting framework whose algorithm is based on decision trees. Through boosting, the trees are constructed sequentially, with each one (i.e., weak learners) aimed at reducing the errors of the previous one. With the help of these weak learners, the boosting technique is able to combine these weak learners to produce a strong learner.
- **Ensemble:** This is a hard voting ensemble learning method that combines the ML algorithms, including LR, SVM, DT, MNB, RF and XGB, which is built for better performance.

4.2.2. Advanced ML Models

Developing an automatic fake news detection model is more important than ever before, given how much data a single person can curate daily. Just think of solutions for detecting and categorising social users' tweets on social media to understand the characteristics of fake and real content. Moreover, detecting fake content from text has already shown its importance in the literature, where adding features extracted from text is essential for good performance in fake news classification. Advanced ML and DL methods have proven their effectiveness in the field of NLP. Here, we investigate the power of 10 different advanced ML models—CNNs, BiLSTM, BiGRU, CNN-BiLSTM and CNN-BiGRU, an ensemble of a CNN, BiLSTM and BiGRU—as well as the transformer-based models $BERT_{base}$ and RoBERTa which are used for fake news detection. The workflow of processing advanced ML models is shown in Figure 2. As shown in the figure, the embedding layer (initialised using Word2Vec, GloVe or $BERT_{base}$) was used to embed the data. Then, the resultant representations were fed into the model (CNN, BiLSTM, etc.), followed by a fully connected layer (FC) comprising a single neuron that was activated using a sigmoid function for classification. The models are described below with their experimental set-ups. Table 5 lists the hyperparameters used in the advanced ML models. Binary cross-entropy and an Adam optimiser were used to train the models:

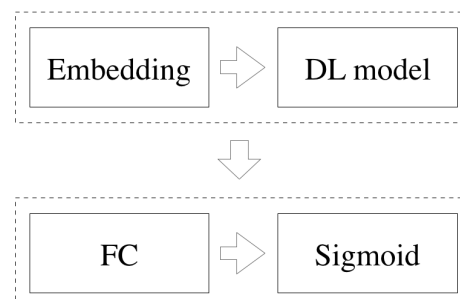


Figure 2. The structure of advanced ML models.

- **CNN:** A one-dimensional convolutional neural network is a powerful ML approach for automatically extracting features from text inputs. A CNN can extract local features

automatically but is less computationally expensive than other ML algorithms. Here, the architecture includes a single CNN layer with 128 filters with a kernel size of 5, which are activated with ReLU as an activation function. The generated feature map is then refined and reduced using the max-pooling layer, resulting in the most relevant information. After that, the output is flattened and passed to a dense output layer, with a single unit activated with a sigmoid as an activation function.

- **LSTM:** The LSTM model [111] improves on the RNN's flaws by adding an additive and multiplicative interaction to the recurrence formula and a distinct memory state. A model's complexity can also be increased by stacking LSTM layers. With three gates—an input gate, a forget gate and an output gate—LSTM models eliminate the gradient vanishing and explosion concerns brought by RNNs. An important characteristic of LSTM models is their ability to capture long-term dependency. The LSTM method has been proven to be effective when used for long sentences [112]. Mathematically, the LSTM components can be formulated as follows [113]:

$$f_t = \sigma(W_{xf} \cdot x_t + W_{hf} \cdot h_{t-1} + b_f), \quad (4)$$

$$i_t = \sigma(W_{xi} \cdot x_t + W_{hi} \cdot h_{t-1} + b_i), \quad (5)$$

$$\tilde{C}_t = \tanh(W_{xc} \cdot x_t + W_{hc} \cdot h_{t-1} + b_i), \quad (6)$$

$$C_t = f_t \odot C_{t-1} + i \odot \tilde{C}_t, \quad (7)$$

$$o_t = \sigma(W_{xo} \cdot x_t + W_{ho} \cdot h_{t-1} + b_o), \quad (8)$$

$$h_t = o_t \odot \tanh(C_t). \quad (9)$$

In the formulas above, σ represents the logistic sigmoid activation function. W , b and C_t represent the weight matrix, the bias and the state of the memory unit at time t , respectively. Here, a single BiLSTM layer with 128 units is used to encode the input text.

- **GRU:** In a GRU variant, there are only two gates: an update gate and a reset gate. The update gate combines the forget and input gates and decides what information will be passed to the current state. The reset gates determine when to ignore the previously hidden state [114]. As with LSTM, the update and reset gates are computed as follows [114]:

$$r_t = \delta(W_r h_{t-1} + U_r x_t + b_r), \quad (10)$$

$$z_t = \delta(W_z h_{t-1} + U_z x_t + b_z), \quad (11)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t, \quad (12)$$

$$\tilde{h}_t = \tanh(W_{\tilde{h}}(h_{t-1} \odot r_t) + U_{\tilde{h}} x_t). \quad (13)$$

In the formulas above, $\delta(\cdot)$ denotes the logistic sigmoid function and W and U show the weight matrices of gates h_t and b , respectively, referring to the hidden state and bias vectors. A basic RNN considers the context of the past but cannot consider the context of the future. Hence, to account for future and previous contexts, bidirectional LSTM (BiLSTM) and bidirectional GRU (BiGRU) are excellent choices thanks to their breakthrough designs. To accomplish this, the forward and backward hidden layers are combined, thereby controlling the temporal information flow in both directions and leading to better learning.

Here, we used a single BiGRU layer with 128 neurons. Even though BiLSTM and BiGRU have shown their superiority in a suite of NLP problems, they are not free from two shortcomings: (1) as the high-dimensional input space increases, so does the complexity of these models, leading to further complexity in optimizing such models, and (2) as these models can capture succeeding and proceeding contextual information (bidirectionality concept), they are not able to focus on the most salient parts of the contextual information of the text. Therefore, to overcome the former issue,

a CNN can be used to reduce the dimensionality of the feature space while retaining the informative features from the text. In addition, a CNN can capture and extract local patterns.

- **CNN-BiLSTM:** Hybridising recurrence-based models with a CNN helps extract salient features, capturing local contextualised patterns and improving the model's accuracy. First, a single CNN layer of 128 filters with a kernel size of 5 is used to process the input vectors and extract the local features. The resultant feature maps of the CNN layer are then fed to a single BiLSTM layer with 128 units to learn the long-term dependencies of the local features of news articles. This is followed by an output layer with a single unit activated with a sigmoid function. The temporal and contextual features and long-term dependencies of the input text can be learned and captured from the text by using an RNN, and important local features can be detected by harnessing the power of the CNN in handling the spatial relations [115,116].
- **CNN-BiGRU:** Similar to the CNN-BiLSTM model, the architecture with a BiLSTM layer was replaced with a BiGRU layer.
- **Hybrid:** This is a hybrid model that combines three models: a single CNN layer with 128 neurons of a kernel size of 5, followed by a max-pooling and then a BiLSTM layer with 128 units and then a BiGRU layer with 128 units.

Table 5. The hyperparameters of the advanced ML models.

Model	Hyperparameters				
	Hidden Layer(s)	Activation Function	Neurons or Filters	Kernel Size	Dropout
CNN	1	ReLU	128	5	0.3
BiLSTM	1	N/A	128	N/A	0.3
BiGRU	1	N/A	128	N/A	0.3

4.2.3. Transformer-Based Models

By analysing existing related works, only a few studies have used transformer-based models to detect fake news, and little research has explored how to best harness such models to detect fake news. It becomes prohibitively challenging to process massive amounts of user-generated content manually. Therefore, automated systems capable of detecting fake content are essential. However, fake news on social media is a non-trivial task, since fake news is written deliberately to mislead readers, and user-generated content is typically of poor quality. To address these challenges, researchers proposed various methods for interpreting the meaning of a word through embedding vectors. Neural network-based methods such as Word2Vec and GloVe are commonly used to learn word embeddings from large word corpora. However, these embedding models have the disadvantage of being context-free since context is neglected, and static embeddings for words are generated regardless of their contexts. Therefore, to achieve finer-grained performance, a model must be able to capture semantic and contextual patterns. Moreover, the advanced ML models can automatically extract semantic information from a given input to detect fake content, but they cannot accurately recognise fake content without a deep understanding of the text. As such, there has been growing interest in the attention paradigm in recent years.

There is an overall paradigm shift taking place in the NLP community, which aims to develop a set of models that not only improve accuracy but also address the problem of lacking labelled data, which has been a long-standing issue in the scientific community. In addition, there is an urgent need to detect fake news automatically. However, this is a challenging task, since existing ML models (prior to the advent of transformer models) fail to provide a deeper semantic understanding of text inputs. This has caused NLP research to make great strides by introducing pretrained transformer-based language models. Using PLMs trained on massive unlabeled data for text classification tasks is becoming increasingly popular. To adapt for the downstream task, new neural network layers are layered on top of the pretrained layers in the PLM [117]. Here, a fully contented

(FC) layer is added on top of the PLMs for classification purposes. A sophisticated approach is needed to detect fake news, since it has become increasingly difficult to distinguish between fake and real content.

This section evaluates two models—BERT [74] and RoBERTa [78]—which have been considered the key breakthroughs of the impressive performance in a suite of NLP tasks, largely due to their powerful language representations being learned from massive amounts of a text corpus. Moreover, such models can be easily fine-tuned to a specific downstream task through so-called transfer learning:

- **BERT:** Originally introduced by Devlin et al. [74], BERT stands for Bidirectional Encoder Representation from Transformers. As the first deeply bidirectional and unsupervised language representation, this model uses a multi-layer bidirectional transformer encoder that simultaneously conditions the left and right contexts. As a result, BERT generates embeddings that are context-aware. BERT further eliminates the unidirectional constraint by performing pretraining using an unsupervised prediction task that includes a masked language model (MLM) which understands context and predicts words. A vector representation can therefore be generated by the model that captures the general information of the input text. The semantic representation of each word in the input text can be improved using an attention mechanism by boosting semantic representation based on the context of the word. The attention mechanism plays an important role in transformer architecture in that it assigns varying weights to different parts of text according to their contributions to the output. An Attention function maps queries and follows key-value and output-vector pairs, which can be seen in Equation (14):

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (14)$$

where Q , K and V denote the query, key and value itself, respectively. $\sqrt{d_k}$ denotes the dimension of the key vector k and query vector q . Attention uses a Softmax activation function that normalises the inputs to a value between 0 and 1. BERT uses a multi-head attention mechanism which can be seen in Equation (15), where each specific head and the associated weight matrices are denoted with the subscript i :

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O \quad (15)$$

where W^O denotes a weight matrix that was trained jointly with the model and each head_i is calculated as follows:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \quad (16)$$

Even though BERT contains millions of parameters (i.e., BERT_{base} contains 110 million parameters, while BERT_{large} has 340 million parameters) [74], in contrast to pretraining, BERT is relatively inexpensive to apply to downstream tasks using jointly fine-tuned parameters based on a pretrained model. In this work, we use BERT_{base}.

- **RoBERTa [78]:** An optimised version of the BERT approach was introduced by Facebook. In this method, BERT is retrained with an improved training methodology by (1) removing the Next Sentence Prediction task from pretraining; (2) using 10 times as much data as BERT to train RoBERTa and (3) introducing dynamic masking with larger batch sizes so that the masked tokens change during training, as opposed to the static masking pattern used in BERT. In this way, RoBERTa differs from BERT in the way it approaches pretraining. We experiment with RoBERTa_{base} in this paper.

4.3. Preprocessing

Preprocessing of the input text included stop word and punctuation removal. The text was then passed through a process of tokenising it into tokens and building a dictionary

as a result of learning the vocabulary included in the given text corpus. Such a dictionary was then used to map every token into a unique integer, resulting in sequences of integers. Then, we padded or truncated such sequences to a fixed number of entries, as we needed to feed the models with vectors of the same length. The downside here was that any vectors that were too long would be truncated, resulting in the loss of some (useful) information. These sequences would then be transformed into fixed-length vectors of word embeddings (i.e., the dimension of such word vector embeddings being used). We then initialised the neural networks with (1) context-independent pretrained word-embedding models such as Word2Vec and GloVe and (2) the context-informed pretrained language model BERT_{base}. For classical ML algorithms, in order to convert tokenised texts into features, CountVectorizer (CV) and TF-IDF were used as statistical feature extractors for the ML models. Additionally, we used pretrained Word2Vec and GloVe to produce word representations as features for classical ML algorithms.

4.4. Experimental Set-Up

All the work for the experiments was carried out using an Intel Core i5 2.3 GHz, 8 GB RAM system running macOS. Classical ML classifiers were implemented using the scikit-learn package and Keras library to implement the advanced ML models. The HuggingFace library was used to implement the PLMs. For the advanced ML models, we carried out the experiments with two scenarios: (1) with context-independent embedding models such as Word2Vec and GloVe and (2) with context-informed embedding models such as BERT_{base}. In our final configuration, we used an Adam optimiser with a default learning rate value for the advanced ML models, namely 2×10^{-5} for BERT_{base} models and 1×10^{-5} for training RoBERTa_{base} models. Again, based on a trial-and-error examination, these values performed the best. For the advanced ML models and PLMs, the sigmoid function was used in the output layer to reduce the error during training, while binary cross-entropy was used to calculate the loss during backpropagation. All models were trained for 10 epochs with a batch size of 32.

4.5. Evaluation Metrics

We employed four evaluation criteria extensively used in text classification tasks, namely accuracy, precision, recall and F1 score (calculated as in the equations below), to assess the performance of the models:

- Accuracy (A): Accuracy is a measure of the classifier's ability to correctly classify a piece of information as either fake or real. The accuracy can be estimated using Equation (17):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (17)$$

- Precision (P): Precision is a measure for the classifier's exactness such that a low value indicates a large number of false positives. The precision represents the number of positive predictions divided by the total number of positive class values predicted and is calculated using Equation (18):

$$Precision = \frac{TP}{TP + FP} \quad (18)$$

- Recall (R): Recall is considered a measure of a classifier's completeness (e.g., a low value of recall indicates many false negatives), where the number of true positives is divided by the number of true positives and the number of false negatives, as can be clearly seen in Equation (19):

$$Recall = \frac{TP}{TP + FN} \quad (19)$$

- F1 score (F1): The F1 score is calculated as the weighted harmonic mean of the precision and recall measures of the classifier using Equation (20):

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \quad (20)$$

where TP , TN , FP and FN are true positive, true negative, false positive and false negative, respectively.

4.6. Datasets

Our study made use of the following datasets described below. These datasets best reflect the real world, where the percentage of fake news that is fabricated and shared is smaller than the volume of news produced on a daily basis. The baseline models on the LIAR, FakeNewsNet-PolitiFact, FakeNewsNet-GossipCop and COVID-19 datasets are summarised in Tables 1–4, respectively.

4.6.1. LIAR

This is a large-scale, publicly available fake news dataset proposed in [69]. The dataset has about 12,800 entries and contains two main components: user profiles and short political statements. The user profile features include the speaker’s name, job, party affiliation, state, credit history and context. The statements (reported during the time interval from 2007 to 2016) have been labelled by the editors of [Politifact.com](https://www.politifact.com) (accessed on 18 March 2022) using six fine-grained categories, namely true, mostly true, half true, barely true, false and pants on fire. These six labels are relatively balanced in size. Overall, each statement has its associated label and information about the speaker of that statement. The authors considered a multi-class classification task on the Liar dataset. Several textual and metadata elements, such as the speaker’s affiliation and the source newspaper, were included, and labels were assigned based on the six degrees of truth that PolitiFact provides. In our study, we cast the problem as a binary classification problem where we set true, mostly true, and half true to “true” and the rest to “false”. The statistics of the dataset are shown in Table 6.

Table 6. The statistics of the LIAR dataset.

# Candidate News	12,791
# True news	7134
# Fake news	5657

4.6.2. FakeNewsNet

This comprehensive dataset (<https://github.com/KaiDMML/FakeNewsNet>) (accessed on 20 March 2022) consists of full-text news articles collected from the [politifact.com](https://www.politifact.com) and [gossipcop.com](https://www.gossipcop.com) (accessed on 18 March 2022) websites. Each of these includes tagged news content (e.g., news articles) and social context information (e.g., relevant social user interactions for news articles). Nevertheless, this contextual information may not always be available in the real world. Thus, we solely relied on content-based features (i.e., news article text). The statistics of the dataset are tabulated in Table 7.

Table 7. The statistics of the FakeNewsNet dataset.

Dataset	PolitiFact	GossipCop
# Candidate news	694	18,676
# True news	356	14,129
# Fake news	338	4547

4.6.3. COVID-19

The goal of the Constraint@AAAI2021 COVID19 fake news detection challenge [118] was to build a model that has the potential to identify whether a message about COVID-19 is fake or real news. This dataset is a collection of COVID-19-related social media posts, comments and news, classified as real or fake based on their truthfulness. The dataset [98] was collected from various social media platforms, such as Twitter and YouTube. In addition, the challenge organisers collected 10,700 social media posts and news articles about COVID-19 in the form of an annotated dataset in English. As the dataset was separated in advance by the task organisers into training, validation and testing sets, we opted to evaluate our models using the original split. The statistics of the dataset are shown in Table 8.

Table 8. The statistics of the COVID-19 dataset.

# Candidate News	6420
# True news	3360
# Fake news	3060

5. Analysis of Experimental Results

Unlike traditional ML algorithms, which require human experts to (manually) encode domain knowledge through feature engineering, which is inefficient and impractical, advanced ML and DL models can learn relevant and important feature representations automatically, making them particularly well-suited for NLP tasks. This section presents the experimental results related to a set of datasets commonly used in fake news detection. First, a comparison of the results of classical ML classifiers using different word representation methods across the used performance metrics is shown in Tables 9, 11 and 13. Second, a comparison of the performance of an assortment of advanced ML models using two scenarios, context-independent embedding models and context-informed embedding models, is presented in Tables 10, 12 and 14.

5.1. Analysis of Results on the LIAR Dataset

This subsection presents the results of an assortment of ML and DL models using the LIAR dataset. It is apparent from Table 9 that the highest accuracy result achieved on the LIAR dataset was 63.9% for the ensemble with TF-IDF features, where this classifier also resulted in significantly high precision and recall values, while the state-of-the-art result achieved on LIAR (binary classification) was 62% (refer to Table 1).

In contrast, the lowest result was 51.78% for DT with GloVe embeddings. The classifier could not detect some patterns in the given input text. As presented in Table 10, the performance of different models was compared. What stands out from this table is that the majority of the models achieved an accuracy of no more than 62% with BERT_{base} and the CNN-BiLSTM using BERT_{base} embeddings reporting an accuracy of 63.06%. It is clear from the table that the lowest result was reported for the recurrence-based models with Word2Vec embeddings. As the LIAR dataset contains short political statements, it is difficult to obtain useful clues that could help discriminate fake news from real news, specifically when using complex models such as LSTM, since this may increase the risk of overfitting. Based on that, we agree with [85] that a news article's information and the length of the dataset are critical factors that affect the performance of recurrence-based models. Such models are more likely to overcome overfitting when sufficient information is provided in a news article. The obtained results indicate that the classical ML methods outperformed sophisticated DL methods on the LIAR dataset, including the most recent advanced PLMs such as BERT_{base} and RoBERTa_{base}.

Table 9. Fake news detection using classical ML classification models on LIAR dataset.

		Dataset			
		LIAR			
Model	Feature	A (%)	P (%)	R (%)	F1 (%)
LR	CV	0.6196	0.6530	0.6933	0.6726
SVM	CV	0.6346	0.6441	0.7857	0.7079
MNB	CV	0.6243	0.6530	0.7115	0.6810
DT	CV	0.5730	0.6137	0.6541	0.6332
RF	CV	0.6275	0.6360	0.7927	0.7057
XGB	CV	0.6093	0.6085	0.8599	0.7127
Ensemble	CV	0.6306	0.6504	0.7451	0.6945
LR	TFIDF	0.6354	0.6493	0.7675	0.8655
SVM	TFIDF	0.6101	0.6421	0.6961	0.6680
MNB	TFIDF	0.6093	0.6311	0.7381	0.6804
DT	TFIDF	0.5414	0.5917	0.6008	0.5962
RF	TFIDF	0.5659	0.5658	0.9874	0.7194
XGB	TFIDF	0.6093	0.6121	0.8375	0.7073
Ensemble	TFIDF	0.6393	0.6503	0.7787	0.7087
LR	Word2Vec	0.6361	0.6462	0.7829	0.7080
SVM	Word2Vec	0.6314	0.6438	0.7745	0.7031
MNB	Word2Vec	0.5762	0.5778	0.9202	0.7099
DT	Word2Vec	0.5493	0.6032	0.5854	0.5942
RF	Word2Vec	0.5675	0.5707	0.9384	0.7097
XGB	Word2Vec	0.6156	0.6300	0.7703	0.6931
Ensemble	Word2Vec	0.6267	0.6355	0.7913	0.7049
LR	GloVe	0.6172	0.6265	0.7941	0.7004
SVM	GloVe	0.6212	0.6306	0.7913	0.7019
MN	GloVe	0.5635	0.5635	0.5635	0.7208
DT	GloVe	0.5178	0.5724	0.5700	0.5712
RF	GloVe	0.6006	0.6042	0.8445	0.7044
XGB	GloVe	0.5943	0.6109	0.7717	0.6819
Ensemble	GloVe	0.5983	0.6099	0.7969	0.6910

5.2. Analysis of Results on FakeNewsNet Dataset

Previous studies used the accuracy, precision, recall and F1 scores to evaluate the performance of the proposed classifiers on the FakeNewsNet dataset. Tables 2 and 3 display the summary of the results obtained from previous work on the PolitiFact and GossipCop datasets, respectively. What is striking about the resulting accuracies in these tables is that relying on modelling the unigram features (weighted by TF-IDF) showed a constantly high accuracy, outperforming the SAF and CNN baselines even with the combinations of social clues and news content on both datasets. As demonstrated in Table 11, the highest accuracy value (89.93%) on the PolitiFact dataset was achieved using the ensemble with CV as a feature extractor method. This model (Ensemble+CV) was found to be the second-best model for the GossipCop dataset, with SVM+TF-IDF (F1 score of 91.55%) being found to be the best-performing model among the other models on the GossipCop dataset. DL approaches, particularly the most recent advanced PLMs, achieved state-of-the-art performance in many NLP tasks. As can be observed from Table 12, RoBERTa_{base} performed the best on the PolitiFact dataset with an F1 score of 93.17, while the accuracy results (92%) were on par with the ensemble advanced ML model using BERT_{base} representations. For the GossipCop dataset, CNN-BERT_{base} was found to be the best-performing model (F1 score of 91.45%). This indicates that context-aware transformer-based models help uncover useful patterns to distinguish fake news from real news. Since the GossipCop dataset is highly imbalanced, different upsampling and downsampling techniques can help balance the dataset and further increase detection performance. By relying only on news text and having minimal text preprocessing, we achieved better results than the state-of-the-art results on the FakeNewsNet dataset.

Table 10. Fake news detection using advanced ML and DL classification models on LIAR dataset.

		Dataset			
		LIAR			
Model	Feature	A (%)	P (%)	R (%)	F1 (%)
CNN	Word2Vec	0.5825	0.6219	0.6611	0.6409
BiLSTM	Word2Vec	0.5533	0.6028	0.6078	0.6053
BiGRU	Word2Vec	0.5572	0.6073	0.6064	0.6069
CNN-LSTM	Word2Vec	0.5635	0.5635	1.0000	0.7208
CNN-GRU	Word2Vec	0.5627	0.5632	0.9986	0.7202
CNN-BiLSTM	Word2Vec	0.5762	0.6281	0.6078	0.6178
CNN-BiGRU	Word2Vec	0.5714	0.6067	0.6807	0.6416
Hybrid	Word2Vec	0.5785	0.6175	0.6625	0.6392
CNN	GloVe	0.6172	0.6388	0.7381	0.6849
BiLSTM	GloVe	0.5888	0.6442	0.6036	0.6233
BiGRU	GloVe	0.5927	0.6371	0.6443	0.6407
CNN-LSTM	GloVe	0.5635	0.5635	1.0000	0.7208
CNN-GRU	GloVe	0.5635	0.5635	1.0000	0.7208
CNN-BiLSTM	GloVe	0.5912	0.6365	0.6401	0.6383
CNN-BiGRU	GloVe	0.6014	0.6240	0.7367	0.6757
Hybrid	GloVe	0.5809	0.6332	0.6092	0.6210
CNN	BERT _{base}	0.5975	0.6555	0.6022	0.6277
BiLSTM	BERT _{base}	0.6204	0.6392	0.7493	0.6899
BiGRU	BERT _{base}	0.6180	0.6386	0.7423	0.6865
CNN-LSTM	BERT _{base}	0.6306	0.6602	0.7101	0.6842
CNN-GRU	BERT _{base}	0.6077	0.6543	0.6443	0.6493
CNN-BiLSTM	BERT _{base}	0.5793	0.6100	0.7031	0.6532
CNN-BiGRU	BERT _{base}	0.6140	0.6313	0.7577	0.6887
Hybrid	BERT _{base}	0.5912	0.6195	0.7115	0.6623
BERT _{base}	BERT _{base}	0.6306	0.6662	0.6905	0.6781
RoBERTa _{base}	RoBERTa _{base}	0.6117	0.6468	0.6849	0.6653

5.3. Analysis of Results on COVID-19 Dataset

BERT can effectively detect fake content as it has the power to encode deep semantic contextual information. Based on the results in Tables 13 and 14, we have shown that the BERT_{base} model, with a linear layer being added on top of it for classification, achieved the best results compared with its counterparts and other classical ML and DL approaches, followed by CNN-BERT_{base}, which was ranked as the second-best-performing model on the COVID-19 dataset. Due to the incapability of vector space methods such as CV and TF-IDF to take context into account, using these representations with ML classifiers relies on the appearance of tokens in making final decisions, regardless of their context. The results of our study indicate that vector space models are ineffective at capturing the deeper semantics and contextual patterns contained in user-generated content on Twitter. A major advantage of BERT (and its variations) in the case of Twitter (where user-generated content often contains misspellings, noise and abbreviations) is the use of sub-tokens rather than fixed tokens. It is thus ideal for use with such data [119] instead of standard context-independent word embeddings that operate at the word level.

Table 11. Fake news detection using classical ML classification models on FakeNewsNet dataset.

		Datasets							
		PolitiFact				GossipCop			
Model	Feature	A (%)	P (%)	R (%)	F1 (%)	A (%)	P (%)	R (%)	F1 (%)
LR	CV	0.8311	0.7374	0.9771	0.8405	0.8375	0.8644	0.9333	0.8976
SVM	CV	0.8330	0.7248	0.9978	0.8397	0.8183	0.8136	0.9881	0.8924
MNB	CV	0.7220	0.5393	1.0000	0.7007	0.8335	0.8417	0.9628	0.8982
DT	CV	0.7419	0.8050	0.7758	0.7901	0.7875	0.8607	0.8607	0.8607
RF	CV	0.8131	0.6918	0.9977	0.8171	0.8536	0.8505	0.9803	0.9108
XGB	CV	0.8548	0.7720	0.9840	0.8652	0.8429	0.8442	0.9737	0.9043
Ensemble	CV	0.8993	1.0000	0.8250	0.9041	0.8528	0.8521	0.9765	0.9100
LR	TFIDF	0.8567	0.9979	0.7642	0.8655	0.8587	0.8605	0.9723	0.9130
SVM	TFIDF	0.8416	0.9855	0.7484	0.8508	0.8635	0.8667	0.9702	0.9155
MNB	TFIDF	0.7524	0.9973	0.5912	0.7423	0.8212	0.8574	0.9182	0.8868
DT	TFIDF	0.6841	0.8272	0.6022	0.6970	0.7861	0.8612	0.8578	0.8595
RF	TFIDF	0.7960	1.000	0.6619	0.7966	0.8496	0.8435	0.9856	0.9090
XGB	TFIDF	0.8748	0.9846	0.8050	0.8858	0.8410	0.8445	0.9702	0.9030
Ensemble	TFIDF	0.8777	0.9565	0.8250	0.8859	0.8611	0.8650	0.9691	0.9141
LR	Word2Vec	0.7884	0.6525	0.9952	0.7882	0.8383	0.8421	0.9698	0.9015
SVM	Word2Vec	0.7922	0.6651	0.9860	0.7944	0.8367	0.8325	0.9839	0.9019
MNB	Word2Vec	0.6575	0.9791	0.4418	0.6089	0.7626	0.7626	1.0000	0.8653
DT	Word2Vec	0.6907	0.5393	0.9122	0.6779	0.7578	0.8529	0.8245	0.8385
RF	Word2Vec	0.6907	0.4984	0.9784	0.6604	0.8394	0.8388	0.9772	0.9027
XGB	Word2Vec	0.6983	0.5047	0.9907	0.6688	0.8332	0.8406	0.9642	0.8982
Ensemble	Word2Vec	0.8849	0.9324	0.8625	0.8961	0.8410	0.8395	0.9786	0.9037
LR	GloVe	0.6992	0.9819	0.5110	0.6722	0.7628	0.7632	0.9989	0.8653
SVM	GloVe	0.7239	0.9859	0.5503	0.7064	0.7626	0.7626	1.0000	0.8653
MN	GloVe	0.6575	0.9758	0.4434	0.6097	0.7626	0.7626	1.0000	0.8653
DT	GloVe	0.6262	0.8151	0.4921	0.6137	0.7230	0.8265	0.8059	0.8161
RF	GloVe	0.6546	0.9789	0.4371	0.6043	0.8236	0.8228	0.9796	0.8944
XGB	GloVe	0.7106	0.9853	0.5283	0.6878	0.8062	0.8137	0.9674	0.8839
Ensemble	GloVe	0.7986	0.9194	0.7125	0.8028	0.7944	0.7912	0.9923	0.8804

Table 12. Fake news detection using advanced ML and DL classification models on FakeNewsNet dataset.

		Datasets							
		PolitiFact				GossipCop			
Model	Feature	A (%)	P (%)	R (%)	F1 (%)	A (%)	P (%)	R (%)	F1 (%)
CNN	Word2Vec	0.6475	0.9189	0.4250	0.5812	0.8415	0.8566	0.9516	0.9016
BiLSTM	Word2Vec	0.7770	0.9153	0.6750	0.7770	0.8239	0.8702	0.9038	0.8867
BiGRU	Word2Vec	0.7554	0.8194	0.7375	0.7763	0.8263	0.8616	0.9200	0.8898
CNN-LSTM	Word2Vec	0.7266	0.7059	0.9000	0.7912	0.8303	0.8700	0.9140	0.8915
CNN-GRU	Word2Vec	0.6691	0.6848	0.7875	0.7326	0.8266	0.8694	0.9091	0.8888
CNN-BiLSTM	Word2Vec	0.7050	0.7532	0.7250	0.7389	0.8260	0.8683	0.9098	0.8886
CNN-BiGRU	Word2Vec	0.7338	0.7417	0.8250	0.7811	0.8097	0.8791	0.8701	0.8746
Hybrid	Word2Vec	0.7050	0.8197	0.6250	0.7092	0.8298	0.8639	0.9221	0.8920
CNN	GloVe	0.8561	0.8750	0.8750	0.8750	0.8097	0.8968	0.8480	0.8717
BiLSTM	GloVe	0.8129	0.8971	0.7625	0.8243	0.8284	0.8548	0.9337	0.8925
BiGRU	GloVe	0.8201	0.8571	0.8250	0.8408	0.8271	0.8580	0.9266	0.8910
CNN-LSTM	GloVe	0.7554	0.8382	0.7125	0.7703	0.8185	0.8709	0.8947	0.8826
CNN-GRU	GloVe	0.7194	0.7356	0.8000	0.7665	0.8137	0.8688	0.8901	0.8793
CNN-BiLSTM	GloVe	0.7986	0.8611	0.7750	0.8158	0.8290	0.8590	0.9280	0.8922
CNN-BiGRU	GloVe	0.8129	0.8375	0.8375	0.8375	0.8180	0.8675	0.8986	0.8828
Hybrid	GloVe	0.8085	0.8354	0.8250	0.8302	0.8349	0.8598	0.9361	0.8963
CNN	BERT _{base}	0.8993	0.9342	0.8875	0.9103	0.8616	0.8684	0.9702	0.9145
BiLSTM	BERT _{base}	0.9065	0.9589	0.8750	0.9150	0.8391	0.8769	0.9179	0.8969
BiGRU	BERT _{base}	0.9065	0.9136	0.9250	0.9193	0.8533	0.8754	0.9417	0.9073
CNN-LSTM	BERT _{base}	0.8561	0.8750	0.8750	0.8750	0.8418	0.8729	0.9277	0.8994
CNN-GRU	BERT _{base}	0.8417	0.8537	0.8750	0.8642	0.8514	0.8784	0.9396	0.9061
CNN-BiLSTM	BERT _{base}	0.8993	0.9125	0.9125	0.9125	0.8370	0.8756	0.9165	0.8956
CNN-BiGRU	BERT _{base}	0.8777	0.9437	0.8375	0.8874	0.8322	0.8915	0.8880	0.8897
Hybrid	BERT _{base}	0.9209	0.9600	0.9000	0.9290	0.8437	0.8719	0.9319	0.9009
BERT _{base}	BERT _{base}	0.9137	0.9359	0.9125	0.9241	0.8544	0.8724	0.9477	0.9085
RoBERTa _{base}	RoBERTa _{base}	0.9209	0.9259	0.9375	0.9317	0.8383	0.8794	0.9133	0.8960

Table 13. Fake news detection using classical ML classification models on COVID-19 dataset.

		Dataset			
		COVID-19			
Model	Feature	A (%)	P (%)	R (%)	F1 (%)
LR	CV	0.9313	0.9363	0.9321	0.9342
SVM	CV	0.9318	0.9333	0.9366	0.9349
MNB	CV	0.9051	0.9127	0.9054	0.9090
DT	CV	0.8874	0.8921	0.8929	0.8925
RF	CV	0.9271	0.9221	0.9402	0.9310
XGB	CV	0.8944	0.8942	0.9054	0.8997
Ensemble	CV	0.9327	0.9444	0.9259	0.9351
LR	TFIDF	0.9294	0.9209	0.9464	0.9335
SVM	TFIDF	0.9430	0.9354	0.9571	0.9462
MNB	TFIDF	0.9187	0.9085	0.9393	0.9236
DT	TFIDF	0.8696	0.8778	0.8723	0.8751
RF	TFIDF	0.8654	0.8467	0.9071	0.8759
XGB	TFIDF	0.8883	0.8930	0.8938	0.8934
Ensemble	TFIDF	0.9360	0.9408	0.9366	0.9387
LR	Word2Vec	0.9028	0.9050	0.9098	0.9074
SVM	Word2Vec	0.9051	0.9018	0.9187	0.9102
MNB	Word2Vec	0.8388	0.8083	0.9071	0.8549
DT	Word2Vec	0.7916	0.7993	0.8036	0.8014
RF	Word2Vec	0.8734	0.8794	0.8786	0.8790
XGB	Word2Vec	0.8916	0.8943	0.8991	0.8967
Ensemble	Word2Vec	0.9023	0.9175	0.8938	0.9055
LR	GloVe	0.8182	0.8170	0.8411	0.8289
SVM	GloVe	0.8173	0.8145	0.8429	0.8284
MN	GloVe	0.6486	0.6024	0.6961	0.7421
DT	GloVe	0.7379	0.7450	0.7589	0.7519
RF	GloVe	0.8280	0.8144	0.8696	0.8411
XGB	GloVe	0.8416	0.8381	0.8643	0.8510
Ensemble	GloVe	0.8449	0.8493	0.8554	0.8523

Table 14. Fake news detection using advanced ML and DL classification models on COVID-19 dataset.

		Dataset			
		COVID-19			
Model	Feature	A (%)	P (%)	R (%)	F1 (%)
CNN	Word2Vec	0.9187	0.9022	0.9473	0.9242
BiLSTM	Word2Vec	0.9257	0.9309	0.9268	0.9289
BiGRU	Word2Vec	0.9294	0.9246	0.9420	0.9332
CNN-LSTM	Word2Vec	0.9290	0.9291	0.9357	0.9324
CNN-GRU	Word2Vec	0.9332	0.9342	0.9384	0.9363
CNN-BiLSTM	Word2Vec	0.9304	0.9362	0.9304	0.9333
CNN-BiGRU	Word2Vec	0.9070	0.9656	0.8527	0.9056
Hybrid	Word2Vec	0.9294	0.9145	0.9545	0.9340
CNN	GloVe	0.9542	0.9554	0.9571	0.9563
BiLSTM	GloVe	0.9355	0.9161	0.9652	0.9400
BiGRU	GloVe	0.9421	0.9220	0.9714	0.9461
CNN-LSTM	GloVe	0.8593	0.7991	0.9768	0.8791
CNN-GRU	GloVe	0.5234	0.5234	0.9991	0.6869
CNN-BiLSTM	GloVe	0.9477	0.9308	0.9723	0.9511
CNN-BiGRU	GloVe	0.9472	0.9382	0.9625	0.9502
Hybrid	GloVe	0.9374	0.9143	0.9714	0.9420
CNN	BERT _{base}	0.9752	0.9725	0.9804	0.9764
BiLSTM	BERT _{base}	0.9650	0.9485	0.9866	0.9672
BiGRU	BERT _{base}	0.9706	0.9616	0.9830	0.9722
CNN-LSTM	BERT _{base}	0.9598	0.9434	0.9821	0.9624
CNN-GRU	BERT _{base}	0.9626	0.9710	0.9571	0.9640
CNN-BiLSTM	BERT _{base}	0.9621	0.9529	0.9759	0.9643
CNN-BiGRU	BERT _{base}	0.9565	0.9485	0.9696	0.9589
Hybrid	BERT _{base}	0.9617	0.9642	0.9625	0.9634
BERT _{base}	BERT _{base}	0.9771	0.9735	0.9830	0.9782
RoBERTa _{base}	RoBERTa _{base}	0.9668	0.9541	0.9839	0.9688

6. Discussion

In this section, we will have an overall discussion of the experimental results in the previous subsections. The experiments were carried out using a set of datasets commonly used in fake news detection by applying an assortment of classical and advanced ML classifiers using different word representation methods as well as different transformer-based architectures. The results of the models on the LIAR dataset are tabulated in Tables 9 and 10. The TF-IDF-based method for extracting text features performed better, with the ensemble approach exhibiting the highest accuracy while the CNN-LSTM/CNN-GRU hybrid model with GloVe embeddings had the highest F1 score. Despite their ability to capture automatic features, advanced ML models may produce unsatisfactory results when used individually at this level of analysis, where hybrid algorithms generally perform better. For instance, when a CNN and LSTM were combined with BERT_{base}, the classification accuracy was comparable to that of the best-performing model. As such, we agree with [80] that combining different models based on CNNs and recurrence-based models helps uncover useful patterns to discriminate fake from real content, where a model based on this combination would not only be able to extract the most relevant and salient local features but also capture long-term dependencies. Additionally, BERT_{base} alone achieved an accuracy result that was on par with the best-performing models, demonstrating the power of such a model in capturing useful clues from a short text sample. Moreover, we observed that the CNN network with context-free embedding models beat the recurrence-based models, whereas the opposite was true for the embedding models that took context into account.

Using the FakeNewsNet dataset, as shown in Tables 11 and 12, we found that the transformer-based DL architectures outperformed the state-of-the-art approaches on the PolitiFact dataset. In contrast, classical ML models, such as SVM with TF-IDF features, beat the DL models on the GossipCop dataset, including the official baselines. On the COVID-19 dataset (see Tables 13 and 14), BERT_{base} was ranked as the top-performing model, and CNN-BERT_{base} was the second-best model. The results show that transformer-based models are powerful in capturing the intricacies of such (noisy and sparse) datasets.

7. Conclusions

This paper presents an overview of fake news detection approaches found in the literature and compares seven classical ML algorithms, namely LR, SVM, NB, DT, RF, XGB and an ensemble that combines all such algorithms, with two scenarios of word representation methods: statistical (sparse) word vector representation methods and context-free (dense) pretrained word representation models. In addition, this paper compares eight advanced ML models, namely the CNN, BiLSTM, BiGRU, CNN-BiLSTM, CNN-BiGRU and different hybrid models with two types of text-representation models—context-free and context-aware embedding models—and two advanced pretrained transformer-based models: BERT_{base} and RoBERTa_{base}. We found that the ensemble of the classical ML methods with TF-IDF features outperformed the other methods on the LIAR dataset, including advanced ML models. As the LIAR dataset contains short political statements, it is difficult to obtain useful clues that could help discriminate fake news from real news. However, BERT_{base} alone was able to achieve an accuracy result that was on par with the best-performing models, demonstrating the power of such a model in capturing useful clues from a short piece of text. RoBERTa_{base} performed the best on the PolitiFact dataset with an F1 score of 93.17. In contrast, classical ML methods, such as SVM with TF-IDF features, beat the DL models on the GossipCop dataset, including the official baselines. For the COVID-19 dataset, the best-performing model was BERT_{base}. By relying only on news text with minimal text preprocessing, we achieved better results than the state-of-the-art results across the used datasets. It may be possible to improve the performance if context-based information, in addition to news text, is considered and if other factors, such as style and sentiment, are included. Generally, the experimental results indicate that no single technique can deliver the best performance scores across all used datasets.

To this end, we hope our comparative study will contribute to future research in this area and identify the best model for detecting fake news.

Author Contributions: Conceptualisation, experiments and formal analysis, J.A.; paper writing, J.A.; writing—review and editing, Y.L.; supervision, S.L. and Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The LIAR data is available from [69], FakeNewsNet data is available from <https://github.com/KaiDMML/FakeNewsNet> (accessed on 20 March 2022) and COVID-19 data is available from [118].

Conflicts of Interest: The authors declare no conflict of interest.

References

- Bondielli, A.; Marcelloni, F. A survey on fake news and rumour detection techniques. *Inf. Sci.* **2019**, *497*, 38–55. [CrossRef]
- Kwak, H.; Lee, C.; Park, H.; Moon, S. What is Twitter, a Social Network or a News Media? In *Proceedings of the 19th International Conference on World Wide Web*; Association for Computing Machinery: New York, NY, USA, 2010; pp. 591–600. [CrossRef]
- Allcott, H.; Gentzkow, M. Social media and fake news in the 2016 election. *J. Econ. Perspect.* **2017**, *31*, 211–236. [CrossRef]
- Shu, K.; Sliva, A.; Wang, S.; Tang, J.; Liu, H. Fake News Detection on Social Media: A Data Mining Perspective. *arXiv* **2017**, arXiv:1708.01967.
- Trends, G. “Fake News—Explore—Google Trends”. Available online: <https://trends.google.com/trends/explore?date=2010-01-01%202022-07-14&q=%22fake%20news%22> (accessed on 20 July 2022).
- Langin, K. Fake news spreads faster than true news on Twitter—Thanks to people, not bots. *Sci. Mag.* **2018**. Available online: <https://www.science.org/content/article/fake-news-spreads-faster-true-news-twitter-thanks-people-not-bots> (accessed on 20 February 2022).
- Zubiaga, A.; Aker, A.; Bontcheva, K.; Liakata, M.; Procter, R. Detection and Resolution of Rumours in Social Media: A Survey. *ACM Comput. Surv.* **2018**, *51*, 1–36. [CrossRef]
- Zhao, Z.; Resnick, P.; Mei, Q. Enquiring Minds: Early Detection of Rumors in Social Media from Enquiry Posts. In *Proceedings of the 24th International Conference on World Wide Web*; International World Wide Web Conferences Steering Committee: Geneva, Switzerland, 2015; pp. 1395–1405. [CrossRef]
- Vosoughi, S.; Roy, D.; Aral, S. The spread of true and false news online. *Science* **2018**, *359*, 1146–1151. [CrossRef]
- Kumar, S.; West, R.; Leskovec, J. Disinformation on the web: Impact, characteristics, and detection of wikipedia hoaxes. In *Proceedings of the 25th International Conference on World Wide Web*, Montreal, QC, Canada, 11–15 April 2016; pp. 591–602.
- Friggeri, A.; Adamic, L.; Eckles, D.; Cheng, J. Rumor cascades. In *Proceedings of the International AAAI Conference on Web and Social Media*, Ann Arbor, MI, USA, 1–4 June 2014; Volume 8.
- Zhou, X.; Zafarani, R. A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities. *ACM Comput. Surv.* **2020**, *53*, 1–40. [CrossRef]
- Conroy, N.K.; Rubin, V.L.; Chen, Y. Automatic deception detection: Methods for finding fake news. *Proc. Assoc. Inf. Sci. Technol.* **2015**, *52*, 1–4. [CrossRef]
- Zhou, X.; Jain, A.; Phoha, V.V.; Zafarani, R. Fake News Early Detection: An Interdisciplinary Study. *arXiv* **2019**, arXiv:1904.11679.
- Feng, S.; Banerjee, R.; Choi, Y. Syntactic Stylometry for Deception Detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*; Association for Computational Linguistics: Jeju Island, Republic of Korea, 2012; pp. 171–175.
- Karimi, H.; Tang, J. Learning hierarchical discourse-level structure for fake news detection. *arXiv* **2019**, arXiv:1903.07389.
- Pisarevskaya, D. Deception detection in news reports in the russian language: Lexics and discourse. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing Meets Journalism*, Copenhagen, Denmark, 7 September 2017; pp. 74–79.
- Pérez-Rosas, V.; Kleinberg, B.; Lefevre, A.; Mihalcea, R. Automatic detection of fake news. *arXiv* **2017**, arXiv:1708.07104.
- Chen, Y.; Conroy, N.J.; Rubin, V.L. Misleading Online Content: Recognizing Clickbait as “False News”. In *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection*; Association for Computing Machinery: New York, NY, USA, 2015; pp. 15–19. [CrossRef]
- Potthast, M.; Kiesel, J.; Reinartz, K.; Bevendorff, J.; Stein, B. A stylometric inquiry into hyperpartisan and fake news. *arXiv* **2017**, arXiv:1702.05638.
- Fuller, C.M.; Biros, D.P.; Wilson, R.L. Decision support for determining veracity via linguistic-based cues. *Decis. Support Syst.* **2009**, *46*, 695–703. [CrossRef]

22. Zhou, L.; Burgoon, J.K.; Nunamaker, J.F.; Twitchell, D. Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated communications. *Group Decis. Negot.* **2004**, *13*, 81–106. [CrossRef]
23. Pennebaker, J.W.; Francis, M.E.; Booth, R.J. Linguistic inquiry and word count: LIWC 2001. *Mahway Lawrence Erlbaum Assoc.* **2001**, *71*, 2001.
24. Riedel, B.; Augenstein, I.; Spithourakis, G.P.; Riedel, S. A simple but tough-to-beat baseline for the Fake News Challenge stance detection task. *arXiv* **2018**, arXiv:cs.CL/1707.03264.
25. Ahmed, H.; Traore, I.; Saad, S. Detection of online fake news using n-gram analysis and machine learning techniques. In *Proceedings of the International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 127–138.
26. Bharadwaj, P.; Shao, Z. Fake news detection with semantic features and text mining. *Int. J. Nat. Lang. Comput. (IJNLC)* **2019**, *8*, 17–22. [CrossRef]
27. Wynne, H.E.; Wint, Z.Z. Content based fake news detection using n-gram models. In *Proceedings of the 21st International Conference on Information Integration and Web-Based Applications & Services*, Munich, Germany, 2–4 December 2019; pp. 669–673.
28. Gravanis, G.; Vakali, A.; Diamantaras, K.; Karadais, P. Behind the cues: A benchmarking study for fake news detection. *Expert Syst. Appl.* **2019**, *128*, 201–213. [CrossRef]
29. Burgoon, J.K.; Blair, J.P.; Qin, T.; Nunamaker, J.F. Detecting deception through linguistic analysis. In *Proceedings of the International Conference on Intelligence and Security Informatics*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 91–101.
30. Hancock, J.T.; Curry, L.E.; Goorha, S.; Woodworth, M. On lying and being lied to: A linguistic analysis of deception in computer-mediated communication. *Discourse Process.* **2007**, *45*, 1–23. [CrossRef]
31. Newman, M.L.; Pennebaker, J.W.; Berry, D.S.; Richards, J.M. Lying Words: Predicting Deception from Linguistic Styles. *Personal. Soc. Psychol. Bull.* **2003**, *29*, 665–675. [CrossRef]
32. Tausczik, Y.R.; Pennebaker, J.W. The psychological meaning of words: LIWC and computerized text analysis methods. *J. Lang. Soc. Psychol.* **2010**, *29*, 24–54. [CrossRef]
33. Vrij, A. *Detecting Lies and Deceit: The Psychology of Lying and Implications for Professional Practice*; Wiley: Hoboken, NJ, USA, 2000.
34. Knapp, M.L.; Hart, R.P.; Dennis, H.S. An exploration of deception as a communication construct. *Hum. Commun. Res.* **1974**, *1*, 15–29. [CrossRef]
35. Horne, B.; Adali, S. This just in: Fake news packs a lot in title, uses simpler, repetitive content in text body, more similar to satire than real news. In *Proceedings of the International AAAI Conference on Web and Social Media*, Montreal, QC, Canada, 15–18 May 2017; Volume 11, pp. 759–766.
36. Papadopoulou, O.; Zampoglou, M.; Papadopoulos, S.; Kompatsiaris, I. A two-level classification approach for detecting clickbait posts using text-based features. *arXiv* **2017**, arXiv:1710.08528.
37. Rubin, V.L.; Conroy, N.; Chen, Y.; Cornwell, S. Fake news or truth? using satirical cues to detect potentially misleading news. In *Proceedings of the Second Workshop on Computational Approaches to Deception Detection*, San Diego, CA, USA, 17 June 2016; pp. 7–17.
38. Chowdhury, G.G. Natural language processing. *Annu. Rev. Inf. Sci. Technol.* **2003**, *37*, 51–89. [CrossRef]
39. Qazvinian, V.; Rosengren, E.; Radev, D.R.; Mei, Q. Rumor has it: Identifying Misinformation in Microblogs. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*; Association for Computational Linguistics: Edinburgh, UK, 2011; pp. 1589–1599.
40. Castillo, C.; Mendoza, M.; Poblete, B. Information Credibility on Twitter. In *Proceedings of the 20th International Conference on World Wide Web*; Association for Computing Machinery: New York, NY, USA, 2011; pp. 675–684. [CrossRef]
41. Hamidian, S.; Diab, M.T. Rumor Detection and Classification for Twitter Data. *arXiv* **2019**, arXiv:1912.08926.
42. Yang, F.; Liu, Y.; Yu, X.; Yang, M. Automatic Detection of Rumor on Sina Weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*; Association for Computing Machinery: New York, NY, USA, 2012. [CrossRef]
43. Ajao, O.; Bhowmik, D.; Zargari, S. Sentiment Aware Fake News Detection on Online Social Networks. In *Proceedings of the ICASSP 2019—2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK, 12–17 May 2019; pp. 2507–2511.
44. Ratkiewicz, J.; Conover, M.; Meiss, M.; Gonçalves, B.; Flammini, A.; Menczer, F. Detecting and tracking political abuse in social media. In *Proceedings of the International AAAI Conference on Web and Social Media*, Barcelona, Spain, 17–21 July 2011; Volume 5.
45. Samonte, M.J.C. Polarity analysis of editorial articles towards fake news detection. In *Proceedings of the 2018 International Conference on Internet and e-Business*, Singapore, 25–27 April 2018; pp. 108–112.
46. Volkova, S.; Shaffer, K.; Jang, J.Y.; Hodas, N. Separating facts from fiction: Linguistic models to classify suspicious and trusted news posts on twitter. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 647–653.
47. Bhelande, M.; Sanadhya, A.; Puro, M.; Waldia, A.; Yadav, V. Identifying controversial news using sentiment analysis. *Imp. J. Interdiscip. Res.* **2017**, *3*. Available online: <https://www.semanticscholar.org/paper/Identifying-Controversial-News-using-Sentiment-Bhelande-Sanadhya/23862325ff7b53e7851cd4398553d82cbca483f4> (accessed on 22 May 2022).
48. Qin, Y.; Wurzer, D.; Lavrenko, V.; Tang, C. Spotting Rumors via Novelty Detection. *arXiv* **2016**, arXiv:1611.06322.
49. Kwon, S.; Cha, M.; Jung, K. Rumor Detection over Varying Time Windows. *PLoS ONE* **2017**, *12*, e0168344. [CrossRef]

50. Wei, W.; Wan, X. Learning to identify ambiguous and misleading news headlines. *arXiv* **2017**, arXiv:1705.06031.
51. Chakraborty, A.; Paranjape, B.; Kakarla, S.; Ganguly, N. Stop Clickbait: Detecting and Preventing Clickbaits in Online News Media. *arXiv* **2016**, arXiv:1610.09786.
52. Feng, V.W.; Hirst, G. Detecting Deceptive Opinions with Profile Compatibility. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing; Asian Federation of Natural Language Processing*: Nagoya, Japan, 2013; pp. 338–346.
53. Potthast, M.; Köpsel, S.; Stein, B.; Hagen, M. Clickbait Detection. In *Proceedings of the ECIR, Padua, Italy, 20–23 March 2016*.
54. Gupta, A.; Kumaraguru, P.; Castillo, C.; Meier, P. Tweetcred: Real-time credibility assessment of content on twitter. In *Proceedings of the International Conference on Social Informatics*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 228–243.
55. Bhattacharjee, S.D.; Talukder, A.; Balantrapu, B.V. Active learning based news veracity detection with feature weighting and deep-shallow fusion. In *Proceedings of the 2017 IEEE International Conference on Big Data (Big Data)*, Boston, MA, USA, 11–14 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 556–565.
56. Ribeiro, M.H.; Calais, P.H.; Almeida, V.A.; Meira, W., Jr. “Everything I Disagree With is# FakeNews”: Correlating Political Polarization and Spread of Misinformation. *arXiv* **2017**, arXiv:1706.05924.
57. Popat, K. Assessing the Credibility of Claims on the Web. In *Proceedings of the 26th International Conference on World Wide Web Companion*; International World Wide Web Conferences Steering Committee: Geneva, Switzerland, 2017; pp. 735–739. [[CrossRef](#)]
58. Cardoso Durier da Silva, F.; Vieira, R.; Garcia, A.C. Can machines learn to detect fake news? A survey focused on social media. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*, Maui, HI, USA, 8–11 January 2019.
59. Markowitz, D.M.; Hancock, J.T. Linguistic Traces of a Scientific Fraud: The Case of Diederik Stapel. *PLoS ONE* **2014**, *9*, e105937. [[CrossRef](#)] [[PubMed](#)]
60. Ruchansky, N.; Seo, S.; Liu, Y. CSI: A Hybrid Deep Model for Fake News Detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*; Association for Computing Machinery: New York, NY, USA, 2017; pp. 797–806.
61. Lahby, M.; Aqil, S.; Yafooz, W.M.S.; Abakarim, Y. Online Fake News Detection Using Machine Learning Techniques: A Systematic Mapping Study. In *Combating Fake News with Computational Intelligence Techniques*; Lahby, M., Pathan, A.S.K., Maleh, Y., Yafooz, W.M.S., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 3–37. [[CrossRef](#)]
62. Klyuev, V. Fake news filtering: Semantic approaches. In *Proceedings of the 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Noida, India, 29–31 August 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 9–15.
63. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
64. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
65. Arora, S.; Liang, Y.; Ma, T. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of the International Conference on Learning Representations*, Toulon, France, 24–26 April 2017.
66. Le, Q.; Mikolov, T. Distributed representations of sentences and documents. In *Proceedings of the International Conference on Machine Learning*, PMLR, Beijing, China, 21–26 June 2014; pp. 1188–1196.
67. Zubiaga, A.; Liakata, M.; Procter, R. Exploiting context for rumour detection in social media. In *Proceedings of the International Conference on Social Informatics*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 109–123.
68. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; Dean, J. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2013; Volume 2, pp. 3111–3119.
69. Wang, W.Y. “Liar, Liar Pants on Fire”: A New Benchmark Dataset for Fake News Detection. *arXiv* **2017**, arXiv:1705.00648.
70. Qian, F.; Gong, C.; Sharma, K.; Liu, Y. Neural User Response Generator: Fake News Detection with Collective User Intelligence. In *Proceedings of the IJCAI*, Stockholm, Sweden, 13–19 July 2018; Volume 18, pp. 3834–3840.
71. Goldani, M.H.; Momtazi, S.; Safabakhsh, R. Detecting fake news with capsule neural networks. *Appl. Soft Comput.* **2021**, *101*, 106991. [[CrossRef](#)]
72. Girgis, S.; Amer, E.; Gadallah, M. Deep learning algorithms for detecting fake news in online text. In *Proceedings of the 2018 13th International Conference on Computer Engineering and Systems (ICCES)*, Cairo, Egypt, 18–19 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 93–97.
73. Sarnovský, M.; Maslej-Krešňáková, V.; Ivancová, K. Fake News Detection Related to the COVID-19 in Slovak Language Using Deep Learning Methods. *Acta Polytech. Hung.* **2022**, *19*, 43–57. [[CrossRef](#)]
74. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
75. Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. Deep contextualized word representations. *arXiv* **2018**, arXiv:1802.05365.
76. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving Language Understanding by Generative Pre-Training. 2018. Available online: <https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf> (accessed on 10 August 2022).
77. Müller, M.; Salathé, M.; Kummervold, P.E. Covid-twitter-bert: A natural language processing model to analyse COVID-19 content on twitter. *arXiv* **2020**, arXiv:2005.07503.

78. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
79. Kula, S.; Choraś, M.; Kozik, R. Application of the bert-based architecture in fake news detection. In *Proceedings of the Computational Intelligence in Security for Information Systems Conference*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 239–249.
80. Alghamdi, J.; Lin, Y.; Luo, S. Modeling Fake News Detection Using BERT-CNN-BiLSTM Architecture. In *Proceedings of the 2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR)*, Online, 2–4 August 2022; pp. 354–357. [\[CrossRef\]](#)
81. Aggarwal, A.; Chauhan, A.; Kumar, D.; Mittal, M.; Verma, S. Classification of fake news by fine-tuning deep bidirectional transformers based language model. *EAI Endorsed Trans. Scalable Inf. Syst.* **2020**, *7*, e10. [\[CrossRef\]](#)
82. Jwa, H.; Oh, D.; Park, K.; Kang, J.M.; Lim, H. exBAKE: Automatic fake news detection model based on bidirectional encoder representations from transformers (bert). *Appl. Sci.* **2019**, *9*, 4062. [\[CrossRef\]](#)
83. Baruah, A.; Das, K.A.; Barbhuiya, F.A.; Dey, K. Automatic Detection of Fake News Spreaders Using BERT. In *Proceedings of the CLEF (Working Notes)*, Thessaloniki, Greece, 22–25 September 2020.
84. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2019**, arXiv:1910.01108.
85. Khan, J.Y.; Khondaker, M.T.I.; Afroz, S.; Uddin, G.; Iqbal, A. A benchmark study of machine learning models for online fake news detection. *Mach. Learn. Appl.* **2021**, *4*, 100032. [\[CrossRef\]](#)
86. Elhadad, M.K.; Li, K.F.; Gebali, F. A Novel Approach for Selecting Hybrid Features from Online News Textual Metadata for Fake News Detection. In *Proceedings of the Advances on P2P, Parallel, Grid, Cloud and Internet Computing*; Barolli, L., Hellinckx, P., Natwichai, J., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 914–925.
87. Shu, K.; Mahudeswaran, D.; Wang, S.; Lee, D.; Liu, H. FakeNewsNet: A Data Repository with News Content, Social Context and Spatialtemporal Information for Studying Fake News on Social Media. *arXiv* **2018**, arXiv:1809.01286.
88. Shu, K.; Cui, L.; Wang, S.; Lee, D.; Liu, H. DEFEND: Explainable Fake News Detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 395–405. [\[CrossRef\]](#)
89. Koloski, B.; Perdih, T.S.; Robnik-Šikonja, M.; Pollak, S.; Škrlj, B. Knowledge graph informed fake news classification via heterogeneous representation ensembles. *Neurocomputing* **2022**, *496*, 208–226. [\[CrossRef\]](#)
90. Oriola, O. Exploring N-gram, word embedding and topic models for content-based fake news detection in FakeNewsNet evaluation. *Int. J. Comput. Appl.* **2021**, *975*, 8887. [\[CrossRef\]](#)
91. Sadeghi, F.; Jalaly Bidgoly, A.; Amirkhani, H. Fake News Detection on Social Media Using A Natural Language Inference Approach. *Multimed. Tools Appl.* **2020**. [\[CrossRef\]](#)
92. Gautam, A.; Venkatesh, V.; Masud, S. Fake News Detection System using XLNet model with Topic Distributions: CONSTRAINT@AAAI2021 Shared Task. *arXiv* **2021**, arXiv:2101.11425. [\[CrossRef\]](#)
93. Shifath, S.M.S.U.R.; Khan, M.F.; Islam, M.S. A transformer based approach for fighting COVID-19 fake news. *arXiv* **2021**, arXiv:2101.12027. [\[CrossRef\]](#)
94. Veyseh, A.P.B.; Thai, M.T.; Nguyen, T.H.; Dou, D. Rumor detection in social networks via deep contextual modeling. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Vancouver, BC, Canada, 27–30 August 2019; pp. 113–120.
95. Wani, A.; Joshi, I.; Khandve, S.; Wagh, V.; Joshi, R. Evaluating Deep Learning Approaches for COVID-19 Fake News Detection. In *Combating Online Hostile Posts in Regional Languages during Emergency Situation*; Springer International Publishing: Berlin/Heidelberg, Germany, 2021; pp. 153–163. [\[CrossRef\]](#)
96. Shushkevich, E.; Cardiff, J. TUDublin team at Constraint@ AAI2021–COVID19 Fake News Detection. *arXiv* **2021**, arXiv:2101.05701.
97. Felber, T. Constraint 2021: Machine learning models for COVID-19 fake news detection shared task. *arXiv* **2021**, arXiv:2101.03717.
98. Patwa, P.; Sharma, S.; Pykl, S.; Guptha, V.; Kumari, G.; Akhtar, M.S.; Ekbal, A.; Das, A.; Chakraborty, T. Fighting an infodemic: COVID-19 fake news dataset. In *Proceedings of the International Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 21–29.
99. Goldberg, Y. Neural network methods for natural language processing. *Synth. Lect. Hum. Lang. Technol.* **2017**, *10*, 1–309.
100. Wallach, H.M. Topic modeling: Beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, USA, 25–29 June 2006; pp. 977–984.
101. Damashek, M. Gauging similarity with n-grams: Language-independent categorization of text. *Science* **1995**, *267*, 843–848. [\[CrossRef\]](#) [\[PubMed\]](#)
102. Joachims, T. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. Technical Report, Carnegie-Mellon Univ Pittsburgh pa Dept of Computer Science. 1996. Available online: https://www.cs.cornell.edu/people/tj/publications/joachims_97a.pdf (accessed on 11 January 2022).
103. McCallum, A.; Nigam, K. A comparison of event models for naive bayes text classification. In *Proceedings of the AAI-98 Workshop on Learning for Text Categorization*; Citeseer: State College, PA, USA, 1998; Volume 752, pp. 41–48.
104. Trstenjak, B.; Mikac, S.; Donko, D. KNN with TF-IDF based framework for text categorization. *Procedia Eng.* **2014**, *69*, 1356–1364. [\[CrossRef\]](#)

105. Joachims, T. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 137–142.
106. Harris, Z.S. Distributional structure. *Word* **1954**, *10*, 146–162. [[CrossRef](#)]
107. Salton, G. MCGILL, Michael. In *Introduction to Modern Information Retrieval*; McGraw-Hill, Inc.: New York, NY, USA, 1986.
108. Schütze, H.; Manning, C.D.; Raghavan, P. *Introduction to Information Retrieval*; Cambridge University Press: Cambridge, UK, 2008; Volume 39.
109. Mikolov, T.; Yih, W.t.; Zweig, G. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, GA, USA, 9–14 June 2013; pp. 746–751.
110. Howard, J.; Ruder, S. Universal language model fine-tuning for text classification. *arXiv* **2018**, arXiv:1801.06146.
111. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)]
112. Tang, D.; Qin, B.; Liu, T. Document Modeling with Gated Recurrent Neural Network for Sentiment Classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 17–21 September 2015; pp. 1422–1432.
113. Graves, A. Supervised sequence labelling. In *Supervised Sequence Labelling with Recurrent Neural Networks*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 5–13.
114. Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv* **2014**, arXiv:1406.1078.
115. Zhang, X.; Chen, F.; Huang, R. A combination of RNN and CNN for attention-based relation classification. *Procedia Comput. Sci.* **2018**, *131*, 911–917. [[CrossRef](#)]
116. Zhou, C.; Sun, C.; Liu, Z.; Lau, F. A C-LSTM neural network for text classification. *arXiv* **2015**, arXiv:1511.08630.
117. Gururangan, S.; Marasović, A.; Swayamdipta, S.; Lo, K.; Beltagy, I.; Downey, D.; Smith, N.A. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv* **2020**, arXiv:2004.10964.
118. Patwa, P.; Bhardwaj, M.; Guptha, V.; Kumari, G.; Sharma, S.; Pykl, S.; Das, A.; Ekbal, A.; Akhtar, M.S.; Chakraborty, T. Overview of constraint 2021 shared tasks: Detecting english COVID-19 fake news and hindi hostile posts. In *Proceedings of the International Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 42–53.
119. Horne, L.; Matti, M.; Pourjafar, P.; Wang, Z. GRUBERT: A GRU-Based Method to Fuse BERT Hidden Layers for Twitter Sentiment Analysis. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*; Association for Computational Linguistics: Suzhou, China, 2020; pp. 130–138.