WELFake: Word Embedding Over Linguistic Features for Fake News Detection

Pawan Kumar Verma, *Member, IEEE*, Prateek Agrawal, Ivone Amorim, and Radu Prodan, *Member, IEEE*

***Abstract*— Social media is a popular medium for the dissemination of real-time news all over the world. Easy and quick information proliferation is one of the reasons for its popularity. An extensive number of users with different age groups, gender, and societal beliefs are engaged in social media websites. Despite these favorable aspects, a significant disadvantage comes in the form of fake news, as people usually read and share information without caring about its genuineness. Therefore, it is imperative to research methods for the authentication of news. To address this issue, this article proposes a two-phase benchmark model named WELFake based on word embedding (WE) over linguistic features for fake news detection using machine learning classification. The first phase preprocesses the data set and validates the veracity of news content by using linguistic features. The second phase merges the linguistic feature sets with WE and applies voting classification. To validate its approach, this article also carefully designs a novel WELFake data set with approximately 72 000 articles, which incorporates different data sets to generate an unbiased classification output. Experimental results show that the WELFake model categorizes the news in real and fake with a 96.73% which improves the overall accuracy by 1.31% compared to bidirectional encoder representations from transformer (BERT) and 4.25% compared to convolutional neural network (CNN) models. Our frequency-based and focused analyzing writing patterns model outperforms predictive-based related works implemented using the Word2vec WE method by up to 1.73%.**

***Index Terms*— Bidirectional encoder representations from transformer (BERT), convolutional neural network (CNN), fake news, linguistic feature, machine learning (ML), text classification, voting classifier, word embedding (WE).**

I. Introduction

NOWADAYS people around the world are getting much involved on online social networks regardless of age, community, or sex [1]. Communicating using social networks is simple, fast, and attractive to share and transfer information. Currently, social network sites like Facebook trailed by Twitter are the market pioneers, facilitating over 1.3 billion clients with a dynamic monthly variation of 300 million users in average [2]. Their collaborations generate Terabytes of information every second [3], [4]. Online social networks are attractive because of the simple and convenient way to access and circulate information with other people. However, the fast scattering of data at a high rate with minimal effort enables the widespread of false information, such as fake news, which are harmful to society and people.

Fake news are low-quality information with purposefully false data, propagated by individuals or bots that deliberately manipulate message for tattle or political plans. Schudson and Zelizer [5] claimed that the term "fake news" originated in previous centuries together with the mass media itself. Nevertheless, this term attracted increased attention after the U.S. presidential elections of 2016, when the propagation of fake news on social media pulled the attention of a larger number of online users than traditional newsreaders. In the last five months before the elections, approximately 7.5 million tweets contained a link to exceptionally one-sided or false news websites. An interesting and worrying aspect is that false and unsubstantiated news from doubtful sources attracts more audiences than credible information [6]. Relevant work on this topic concluded that fake news spread quicker, penetrate further, and have a deeper impact than true news [7]. There are numerous cases where people accept and spread news without checking their correctness certified by sources. By doing this, they become part of a group that deliberately or unintentionally propagates fake news. The intention behind the proliferation of fake news may be manipulation of public views for financial or political benefit, or simply fun. The negative consequences of this phenomenon are, therefore, undeniable, ranging from wrong decision-making to episodes of bullying and violence. Fig. 1(a) and (b) shows two common examples of fake news over social networks.

False information categories are fake news, satire, misinformation, rumor, hoax, disinformation, propaganda, and opinion spam [8]. These categories are not mutually exclusive, but many researchers used them with different storylines. Although there exist a few websites to check the authenticity of the news like PolitiFact [9], The Washington Post Fact Checker [10], FactCheck [11], Snopes [12], TruthOrFiction [13], FullFact [14], HoaxSlayer [15], Vishvas News [16], Factly Media & Research [17] yet, these websites are unable to spontaneously react to any fake news event [18].

Manuscript received October 2, 2020; revised January 26, 2021 and February 25, 2021; accepted March 18, 2021. This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program through ARTICONF Project under Grant 825134. *(Corresponding author: Prateek Agrawal.)*

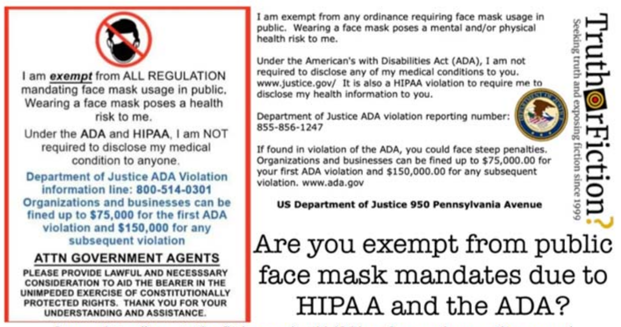
Pawan Kumar Verma is with the Department of Computer Engineering and Applications, GLA University, Mathura 281406, India, and also with School of Computer Science and Engineering, Lovely Professional University, Phagwara 144411, India (e-mail: pawankumar.verma@gla.ac.in).

Prateek Agrawal is with the School of Computer Science and Engineering, Lovely Professional University, Phagwara 144411, India, and also with the Institute of Information Technology, University of Klagenfurt, 9020 Klagenfurt, Austria (e-mail: prateek.agrawal@lpu.co.in).

Ivone Amorim is with MOG Technologies, 4470-605 Moreira, Portugal, and also with the CMUP Mathematical Research Center, University of Porto, 4099-002 Porto, Portugal (e-mail: ivone.amorim@mog-technologies.com).

Radu Prodan is with the Institute of Information Technology, University of Klagenfurt, 9020 Klagenfurt, Austria (e-mail: radu@itec.aau.at).

Digital Object Identifier 10.1109/TCSS.2021.3068519



I am *exempt* from ALL REGULATION mandating face mask usage in public. Wearing a face mask poses a health risk to me.

Under the ADA and HIPAA, I am NOT required to disclose my medical condition to anyone.

Department of Justice ADA Violation information line: 800-514-0301 Organizations and businesses can be fined up to $75,000 for the first ADA violation and $150,000 for any subsequent violation

ATTN GOVERNMENT AGENTS

PLEASE PROVIDE LAWFUL AND NECESSSARY CONSIDERATION TO AID THE BEARER IN THE UNIMPEDED EXERCISE OF CONSTITUTIONALLY PROTECTED RIGHTS. THANK YOU FOR YOUR UNDERSTANDING AND ASSISTANCE.

I am exempt from any ordinance requiring face mask usage in public. Wearing a face mask poses a mental and/or physical health risk to me.

Under the American's with Disabilities Act (ADA), I am not required to disclose any of my medical conditions to you. www.justice.gov/ It is also a HIPAA violation to require me to disclose my health information to you.

Department of Justice ADA violation reporting number: 855-856-1247

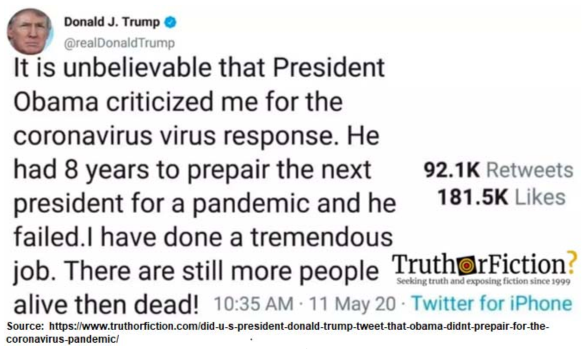
If found in violation of the ADA, you could face steep penalties. Organizations and businesses can be fined up to $75,000.00 for your first ADA violation and $150,000.00 for any subsequent violation. www.ada.gov

US Department of Justice 950 Pennsylvania Avenue

Are you exempt from public face mask mandates due to HIPAA and the ADA?

Source: https://www.truthorfiction.com/covid-19-hipaa-face-mask-exemption-passes/

(a)



Donald J. Trump
@realDonaldTrump

It is unbelievable that President Obama criticized me for the coronavirus virus response. He had 8 years to prepair the next president for a pandemic and he failed.I have done a tremendous job. There are still more people alive then dead! 10:35 AM · 11 May 20 · Twitter for iPhone

92.1K Retweets
181.5K Likes

Source: https://www.truthorfiction.com/did-u-s-president-donald-trump-tweet-that-obama-didnt-prepair-for-the-coronavirus-pandemic/

(b)

Fig. 1. Fake news examples. (a) Decontextualized news. (b) False news.

As online social networks are major sources of information that can mislead individuals or communities [19], there is a serious need for solutions to verify the authenticity of the content. Many researchers consistently try to develop machine learning (ML) models with different sets of features targeted toward automating the fake news detection process [20], [21] using visual [22] or text-based linguistic approaches. However, the following four questions remain unanswered.

1) Which linguistic features are most significant in classifying the news data into real and fake?
2) Which word embedding (WE) technique with linguistic features predicts fake news better than other ML methods like convolutional neural networks (CNNs) [21] or bidirectional encoder representations from transformers (BERTs) [23]?
3) Which classification method is the most appropriate for fake news detection on available data sets?
4) Does ensemble voting classifier improve the fake news detection results?

To answer these questions, we propose a new method called WELFake exclusively focused on text data in three stages.

1) Fake news prediction using *linguistic feature* sets (LFS);
2) *WE* over LFS for improved fake news detection over a WELFake data set.
3) *Comparative analysis* of the linguistic features based results with state-of-the-art CNN and BERT methods.

The WELFake model does not require additional metadata information related to the user or media [24] for the classification of real and fake news. Instead, it aims for a reformation of the state-of-the-art techniques in the detection of fake news over social media websites by using a combined LFS and WE technique. We highlight three contributions of our WELFake model.

A. WELFake Data Set [25]

We designed a larger WELFake data set to prevent overfitting of classifiers and enable better ML training. For this purpose, we merged four popular news data sets (i.e., Kaggle, McIntire, Reuters, and BuzzFeed Political) and prepared a more generic data set of 72 134 news articles with 35 028 real and 37 106 fake news.

B. WELFake Fake News Detection Model

We proposed a novel WELFake model for fake news detection in two steps.

1) collection of various linguistic features from state-of-the-art methods and identification of a subset that performs well on the larger WELFake data set, and
2) ensemble learning on WE features using various ML methods.

C. WELFake Model Generalization and Validation

We applied an adversarial approach to evaluate the model generalization and effectiveness by training and testing on separate data sets.

Experimental results on the WELFake data set revealed that our model achieved a fake news classification accuracy of up to 96.73%, which improves the state of the art by 4.25% over CNN and by 1.31% over BERT methods.

The article has eight sections. Section II gives a background overview on text classification methods. Section III highlights the related work. Section IV describes the proposed methodology and the WELFake model, followed by the resulting algorithm in Section V. Section VI describes the implementation of CNN and BERT state-of-the-art methods for fake news classification used for direct comparison. Section VII provides evaluation results and further related work comparison. Section VIII concludes the article and highlights the future work.

II. Text Classification Background

This section discusses several ML [26] methods, including CNN and BERT for text classification.

A. ML Classification Methods

We review in this section a few ML methods [26] used for fake news classification in the WELFake model.

1) Naive Bayes: This is a supervised learning algorithm based on Bayes' theorem that gives fast predictions with better accuracy in the domain of sentiment analysis, spam filtration, and text classification?

2) Support Vector Machine: This is a supervised learning algorithm that works for both classification and regression problems. The algorithm finds the best line for set separation and predicts the correct set for new data values.

3) Decision Tree: This is a supervised learning algorithm that classifies the data for both categorical and continuous dependent variables. This classifier uses tree structures to solve a problem by distributing complete data sets into homogeneous ones. Internal nodes, branches and leaf nodes in this tree structure represent the data set, the decision rules and the outcome. There are two attribute selection measures for the best attribute node: information gain and Gini index.

4) Random Forest: This is a supervised learning algorithm based on ensemble learning that ensembles several decision trees (DTs) into a random forest (RF) and calculates the average results. The large number of trees in the RF may increase the model accuracy.

5) K-Nearest Neighbor: This is majorly useful for classification problems based on feature similarity. The algorithm can use any integer value for K based on the problem statement and statistics, and employs the Euclidean, Manhattan, or Hamming metric for calculating the distance between data.

6) Boosting: This connects all base learners sequentially. Initially, it passes a few records to the first base learner (BL_1) (of any model) for training, evaluates all the records on BL_1, and passes the incorrectly classified ones to the second learner (BL_2) for training. BL_2 tests all the records and passes the incorrectly classified ones to the next learner BL_3. This process continues until a specified number of base learners.

7) Bagging: This is known as bootstrap aggregation, this is an ensemble technique that uses multiple base learners and provides different subsets of the original data set to each model for training (bootstrapping). The testing process decides the output based on the majority votes from the different models (aggregation). Apart from different sample sets, one can train the models with different subsets to reduce over-fitting.

B. Text Classification Methods

This section reviews the state-of-the-art CNN and BERT classification methods used in our experimental evaluation.

1) CNN [27]: This is each sentence into words (called tokens) and converts them into vectors using context-based WE methods like GloVe, Word2vec, and FastText. These vectors join together to form a $m \times n$ matrix for a given sentence, where m is the embedding dimension and n is the number of tokens present in the vocabulary. Next, it applies the filters on the 1-D convolution (Conv1D) layer. This convolution kernel works from top to bottom (single directional) because of the same width of filters. The Conv1D later combines its outcomes and applies pooling (maximum, average, and global) that passes the data to the final fully connected layer. This takes care of the output generated by the pool layer and produces the classification decision depending on the loads appointed to each other inside the text.

2) BERT [28]: This is a popular pretrained model for text processing developed by Google, which gives a better sense of the language context compared to unidirectional models in two phases.

1) Bidirectional training of an input and output transformer representation model, comprising:
   a) an encoder reading the text as input, and
   b) a decoder generating the output based on the task; and
2) Popular attention mechanism with a neural network (NN) implementing important features only [29].

BERT takes input text and preprocesses it using tokenization, lemmatization, stopword removal, and text lowering operations. Second, it passes the preprocessed text to the encoding phase where additional token, segment, and positional embedding processes take place. Third, it converts the input into default 768 long embedding vectors and passes them through the encoder layers. Finally, the more accurate information of each token available at the last encoding layer passes to the dense layer for text categorization.

III. State-of-the-Art Survey

This section focuses on reviewing various important fake news detection methods.

A. Clickbaits

Clickbaits a special type of fake content that contains linguistic headlines to attract the readers but do not fulfill their promises. Chen *et al.* [30] scrutinized potential techniques for programed discovery of clickbaits by combining syntax and semantics under textual cues, as well as images and newsreader behavior under nontextual cues. They surveyed the importance of these cues for the identification of fake news but failed to implement a prototype of their methodology. Bourgonje *et al.* [31] proposed a model for clickbait detection that checks the relevance of headlines for article bodies. They used the data set released by the coordinators of the first fake news challenge on stance detection and achieved a significant accuracy of 89.59% using a logistic regression classifier. Rashkin *et al.* [32] compared the language of real news with satire, hoaxes, and propaganda, and discovered the features of fake text and other online sources for the fact-checking.

B. News Credibility

Alrubaian *et al.* [33] analyzed the credibility of 489 330 Twitter accounts using text (sentiment analysis), the user (gender, number of followers), and message (URL, hashtag, number of replies) specific features. They implemented RF, Naive Bayes (NB), DT, and feature-rank NB algorithms with a tenfold cross-validation. Castillo *et al.* [34] detected fake news based on credibility using four feature sets: text-specific (e.g., number of characters, words, question marks, punctuation, and sentiment analysis), user-specific (e.g., verified accounts, number of followers, number of tweets, and account creation time), message-specific (e.g., with URL, retweeted status) and propagation (e.g., root node degree and propagation tree depth). These features applied on a Twitter data set obtained precision and recall in the range between 70% and 80%. Benjamin *et al.* [35] classified the news as real or fake based on stylistic (e.g., syntax, text style, and grammar), complexity (e.g., article title), and psycholinguistic features. Experimental results on three data sets revealed better results with large amounts of data and more in-depth features. Kaliyar *et al.* [36] proposed a deep convolutional neural network for fake news detection network (FNDNet) model and achieved a 98.36% accuracy on the Kaggle data set. However, they did not show results on generalized text [37]. Shu *et al.* [38] discussed the worthiness of real or fake news detected through comments on particular items and achieved 90.4%, respectively, 80.80% accuracy on the PolitiFact and GossipCop data sets. Zellers *et al.* [39] proposed a text generation model named GROVER that uses adversaries to spread more trustworthy disinformation than humans. They investigated several propagated threats and showed that GROVER can be an effective discriminator that outperforms BERT in detecting the generated false news.

C. Linguistic Features

The utilization of linguistic features for fake news detection has been popular since the mid-2000s. Burgoon *et al.* [40] used 16 linguistic features categorized in four classes, which achieved an accuracy of 60.72% using a DT algorithm with 15-fold cross-validation. Vicario *et al.* [41] used different features like text (e.g., number of characters, words, sentences, question marks, and negations), user-specific, and message-specific (e.g., number of replies, likes) to identify hoaxes and fake news on social media using linear regression, logistic regression, support vector machine (SVM), K-nearest neighbor (KNN), and NNs. The validation on an Italian Facebook data set with new features achieved an accuracy of 91% on the linear regression classification algorithm. Pérez-Rosas *et al.* [42] used major linguistic features (e.g., n-grams, punctuation, psycho-linguistic, readability, and syntax) and achieved an accuracy of 76% on two novel data sets covering seven domains. Buntain and Golbeck [43] attempted to classify real or fake Twitter threads in four categories using 45 features: structural (e.g., length, the number of tweets), content (e.g., polarity, subjectivity), temporal, and user (e.g., age, followers, authenticity). They used the CREDBANK [44] and PHEME [45] data sets for training and the BuzzFeed data set for testing with an accuracy of 65.29%. Newman *et al.* [46] applied logistic regression to evaluate 29 features and achieved an accuracy between 52% and 67%. Similarly, Zhou *et al.* [47] used 20 features of nine categories for fake news detection. Ahmed *et al.* [48] used an unclear private version of the continuously updated Kaggle data set for fake news detection and achieved an accuracy of 92%. Shu *et al.* [49] used Politifact and BuzzFeed data sets in their analysis, which are hard to generalize due to their very small size. Gravanis *et al.* [37] compared their model with [48], [49] by combining some articles from four data sets (Kaggle-EXT, McIntire, BuzzFeed, and Politifact) in a new UNBiased data set with 3004 articles and achieved the highest accuracy of 95% using SVM. Table I shows a comparative study of five linguistic feature categories.

1) *Readability index* quantifies the text's complexity (i.e., reading difficulty) based on word length, number of syllables, and sentence length.
2) *Psycho-linguistic* features describe emotions, behaviors, persona, and mindset.
3) *Stylistic* features explain the style of a sentence.
4) *User credibility* features describe user information.

TABLE I
STATE-OF-THE-ART COMPARISON OF LINGUISTIC FEATURES

Linguistic feature	[35]	[41]	[48]	[49]	[37]	WELFake
Readability index	✓	✗	✗	✗	✓	✓
Psycho-linguistic	✓	✗	✗	✓	✓	✓
Stylistic	✓	✓	✗	✗	✓	✓
User credibility	✗	✓	✗	✓	✗	✗
Quantity	✗	✗	✗	✗	✓	✓

TABLE II
FAKE NEWS DETECTION DATA SETS

Dataset	*Articles*	*Categories*
Benjamin Political News [35]	225 stories	Real, fake, satire
Burfoot Satire News [50]	4,233 news	Real, satire
BuzzFeed News [35]	101 news	Real, fake
CREDBANK [44]	60 million tweets	Real, fake
Fake News Challenge [51]	50,000 articles	Agree, disagree, discuss, unrelated
FakeNewsNet [49]	422 news	Real, fake
LIAR [52]	12,800 news	True (mostly, half, barely), false, pants-fire
Reuters [48]	44,898 articles	Real, fake
McIntire [53]	6,335 news	Real, fake
Kaggle [54]	38,729 news	Real, fake

5) *Quantity* features explain sentence information such as the number of words and number of sentences.

D. Open Data Sets

From the large number of data sets available for the study of fake news [55]–[61], we highlight some popular data sets in Table II.

1) *Benjamin Political News* is a very small data set with 75 stories each for real, fake, and satire categories.
2) *Burfoot Satire News* is an unbalanced data set with real and satire categories.
3) *BuzzFeed News* is a very small data set of 101 news.
4) *CREDBANK* consists of incomplete news articles in real and fake events categories.
5) *Fake News Challenge* focuses on individual claims among three categories: disagree, discuss and unrelated.
6) *FakeNewsNet* consists of only 422 news articles with incomplete classification in real and fake categories.
7) *LIAR* contains various hard-to-classify social media posts and speeches due to the lack of verification sources or knowledge bases [62].
8) *Reuters* consists of real and fake news articles from a single source that increases the chances of biased data.
9) *McIntire* consists of real and fake news categories without authentic confirmation from any individual.
10) *Kaggle* consists of real and fake news data without source information.

E. Summary

No single method guarantees the best solution for all data sets. The state-of-the-art approach of Gravanis *et al.* [37] used 57 linguistic features and embed them with a word-to-vector embedding method on UNBiased data set of less than 4000 articles only. They used all features in a single LFS

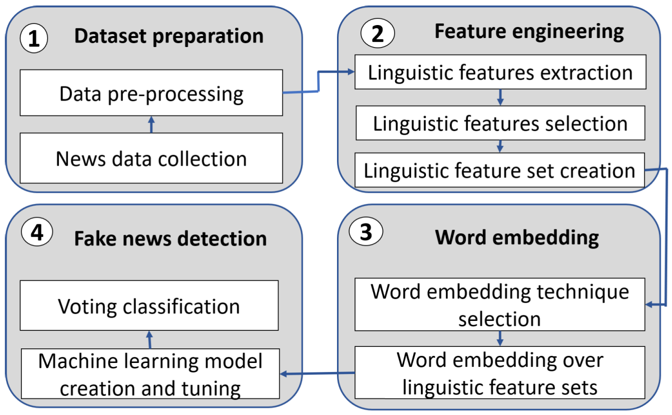

1 Dataset preparation
Data pre-processing
News data collection
2 Feature engineering
Linguistic features extraction
Linguistic features selection
Linguistic feature set creation
3 Word embedding
Word embedding technique selection
Word embedding over linguistic feature sets
4 Fake news detection
Voting classification
Machine learning model creation and tuning

Fig. 2. WELFake model overview.

and achieves up to 95% accuracy. WELFake improves on this method using a novel method based on four stages that:

1) creates a larger data set with improved generalization;
2) identifies the most significant twenty linguistic features and creates three unique LFS based on categories;
3) applies two WE methods to train various ML models; and
4) generates the final prediction using a two-stage voting classification.

IV. WELFake Model

This section outlines the WELFake model for fake news detection divided into four phases, shown in Fig. 2.

1) *Data set preparation* involves the collection and pre-processing of the data in a proper format, as a fundamental task of any ML model;
2) *Feature engineering* involves linguistic feature extraction and selection;
3) *WE* identifies the most appropriate technique connected with the LFS;
4) *Fake news detection* tunes the model parameters and applies a hard voting classifier for better accuracy.

A. Data Set Preparation

1) News Data Collection: This is essential for a balanced and unbiased data set and the key to providing high quality training data and delivering good results. Although there exist an important number of open data sets for the study of fake news (see Section III and Table II), the literature showed their serious limitations in terms of size, category, or bias. After a careful study, we prepared a more comprehensive *WELFake data set* that combines four data sets, Kaggle, McIntire, Reuters, and BuzzFeed, for two reasons. First, they have a similar structure with two categories (i.e., real and fake news). Second, combining the data sets reduces the limitations and the bias of each individual data set. Table III shows the WELFake open data set with 72 134 news articles classified as 35 028 real and 37 106 fake news [25]. The data set contains three columns (i.e., title, text, label) with a binary label for fake and real news. Table IV summarizes the balanced fake and real news distribution in the WELFake data set across all four feature categories.

TABLE III
WELFake Data Set

Dataset	*Real news*	*Fake news*
Kaggle	10387	10413
McIntire	3171	3164
Reuters	21417	23481
BuzzFeed Political	53	48
WELFake dataset	**35,028**	**37,106**

TABLE IV
Fake Versus Real News Distribution in WELFake

Category	*Real news [%]*	*Fake news [%]*
Short sentences	60.9	39.1
Readability index	51.7	48.3
Subjectivity	45.4	54.6
Number of articles	53.9	46.1
WELFake dataset	**48.55%**	**51.45%**

a) Number of short sentences (below ten words) representing real news is greater than those representing fake news.

b) Text readability of fake news is poorer than the readability of real news.

c) Subjectivity of fake news articles is larger than for real news articles.

d) Number of articles representing real news is larger than those representing fake news.

2) Data Preprocessing: This solves different problems in the collected data, like typographic errors, unstructured data format, and other limitations, using several methods, depending on the data set and objectives.

a) Missing data handles undefined (*NaN*) and blank values (*NULL*) present in the data set, which hinder the feature engineering process. Since deleting the data entries containing missing values may cause the loss of important information [63], we performed a missing value imputation process that estimates missing values and then analyzes the complete data set as if these values were the actual observed ones.

b) Inconsistent data deviates from other data points because of mistakes during data collection. We employed several visualization techniques and mathematical functions for outlier identification and correction, like box plot, scatter plot, Z-score, and inter-quartile range (IQR) score.

c) Duplicate data or deduplication removes redundancy that can lead to biased results, which may occur when the same person collected the data.

d) Irrelevant data removes stop words (and other noise) that make the sentence grammatically complete, but do not have semantic significance in news classification operations. Removing stop words and keeping relevant tokens only significantly increases the model performance.

e) Stemming converts the text into its root word by applying the Porter–Stemmer algorithm on text features for accuracy improvement. In case it cannot recognize the root word, it generates the canonical form of a corresponding word.

B. Feature Engineering

1) Linguistic Features Extraction: This is the process of conversion from raw text to the data provided to the ML algorithm. Feature extraction aims to create a feature set that summarizes the information of the original data set, which speeds up the model training and improves the data visualization and learning accuracy. We extracted in WELFake 87 text-based linguistic features from state-of-the-art works, falling into two syntactic (i.e., writing pattern, quantity) and semantic (i.e., grammar, psycho-linguistics) categories.

a) Writing pattern emphasizes the writing style of the text by accessing the sentence types, use of determinants, special characters, and modifiers.

b) Grammar focuses on the text readability index and emphasizes the word structure, average syllables per word, easy word use ratio in a word list, and sentence complexity.

c) Psycho-linguistic estimates the text sentiment and information opinion based on semantic and subjectivity.

d) Quantity identifies speech information parts by counting the verbs, adjectives, adverbs, syllables, and words, as well as the rate of adjectives, adverbs, and words per sentence.

2) Linguistic Features Selection: This is the process of choosing important features for data classification, which diminishes the quantity of input features, decreases the computational expense and improves the accuracy of the ML model. For this purpose, we calculated the Pearson's correlation coefficient of each feature with the other features within the same category and discarded those with a correlation coefficient higher than 0.7, which indicates a strong positive linear relationship [23] between the two features. A learning model with fewer features is simpler and more precise according to Occam's razor principle and the minimum description length concept [62]. We executed this recurrent process until removing all the least significant features. Table V shows the remaining 20 most significant features of WELFake grouped based on the four categories.

3) LFS Creation: This groups the selected 20 linguistic features in different sets to enable multiple WE methods for unbiased model training. We need an odd number of input sets to obtain a clear result upon subsequent voting classification. We consequently created a minimum of three distinct LFS based on four categories, as shown in Table V.

a) Readability index defines the sentence structure complexity of any text. We identify the level or grade of the text writer based on the readability index, which helps in classifying the news as real or fake. For this reason, we uniformly distributed all three readability index features among the three LFS and assigned one feature to each set.

b) Psycho-linguistic features play an important role in detection of fake news, as explained in Section III. We, therefore, used all three of them in all the three LFS.

c) Quantity features also participate in news classification, so we evenly distributed them across the three LFS.

d) Writing pattern contains five features and, therefore, we evenly distributed three features to each LFS.

TABLE V
WELFAKE LFS

Category	*Feature*	*LFS1*	*LFS2*	*LFS3*
Writing pattern	Number of special characters	✓	✗	✓
	Number of determinants	✗	✓	✓
	Number of capital letters	✗	✗	✓
	Number of short sentences	✓	✓	✗
	Number of long sentences	✓	✓	✗
Grammar (readability index)	Gunning fog grade readability index	✓	✗	✗
	SMOG readability index	✗	✓	✗
	Automatic readability index	✗	✗	✓
Psycho-linguistics	Text polarity	✓	✓	✓
	Title similarity	✓	✓	✓
	Subjectivity	✓	✓	✓
Quantity	Number of syllables	✗	✗	✓
	Number of words	✗	✓	✗
	Rate of adjectives and adverbs	✓	✗	✗
	Words per sentence	✗	✗	✓
	Number of articles	✓	✗	✗
	Number of verbs	✗	✓	✗
	Number of sentences	✗	✓	✗
	Number of adjectives	✓	✗	✗
	Number of adverbs	✗	✗	✓

C. WE

1) WE Selection: This identifies the most appropriate technique for converting plain text into a numeric value, as ML methods cannot directly process plain text. We observed two popular WE categories in the literature: 1) *content-based*, such as term frequency-inverse document frequency (TF-IDF) and count vectorizer (CV), that focus on previous knowledge and 2) *context-based*, such as Word2Vec, GloVe, and FastText, that focus on writing text patterns. As fake news writers (fakesters) tend to repeat similar words, we selected the content-based WE that focuses on writing patterns rather than context [64].

a) CV: Also called one-hot encoding, CV converts text document in a histogram vector, where each element represents the number of appearances of the word in the document. The vector length depends on the number of unique words in the corpus.

b) TF-IDF: This is the advanced version of CV, which shows the importance of a term (representing a word) in a corpus alongside its occurrence in the document. *TF-IDF* is the multiplication of the term frequency $tf(t, d)$ that computes the occurrence of a term t in a document d and the inverse document frequency $idf(t, D)$ that computes the importance of that term t in a corpus of documents D

$$\begin{aligned}\text{TF-IDF}(t, d, D) &= tf(t, d) \cdot idf(t, D)\\ tf(t, d) &= \frac{f_d(t)}{\max_{w \in d} f_d(w)}\\ idf(t, D) &= \ln\left(\frac{|D|}{|\{d \in D : t \in d\}|}\right)\end{aligned}$$

where $f_d(t)$ represents the number of occurrences of the term t in the document d, $|D|$ is the number of documents in the corpus D, and $|\{d \in D : t \in d\}|$ is the number of documents containing the term t.

2) WE Over LFS: This improves the output prediction, as predefined features do not always accurately predict and need additional training methods. For this purpose, we combined the WE with LFS. We applied the TF-IDF and CV WE techniques on the three LFS and found that CV gives better results. We achieved maximum of 95.61% accuracy using SVM on CV, while TF-IDF gave maximum accuracy of 95.12% using bagging. Thus, we selected CV and combined it with LFSs for further accuracy analysis of various models in Section VII-B.

TABLE VI
HYPERPARAMETERS FOR ML MODELS TUNING

Model	*Iterations*	*Hyper-parameter*	*Value range*	*Value*
NB	4	Smoothing	$[0, 1]$	1
		Fit prior	{ true, false }	true
SVM	9	Kernel	{ linear, poly, rbf, sigmoid, precomputed }	linear
		Regularization	$\mathbb{R}$	100
		Kernel coefficient	{float, $x \mid x \in$ {scale, auto}}	0.0001
		Degree	$\mathbb{Z}$	3
KNN	5	Number of neighbors	$\mathbb{Z}$	2
		Weights	{ uniform, distance, callable }	uniform
DT	8	Maximum number of features	{integer, float, none, $x \mid x \in$ {auto, sqrt, log 2}}	none
		Criterion	{ gini, entropy }	gini
		Cost complexity pruning	$\mathbb{R}_+$	0.02
Bagging	6	Base estimator	{ object, none }	none
		Number of estimators	$\mathbb{Z}$	100
		Bootstrap	{ true, false }	true
AdaBoost	5	Number of estimators	$\mathbb{Z}$	50
		Learning rate	$[0, 1]$	1

D. Fake News Detection

1) ML Model Creation and Tuning: This passes the LFS with WE through six ML methods: SVM, NB, KNN, DT, Bagging, and AdaBoost. For this purpose, we experimented with each ML model on random samples of the WELFake data set with four training-testing data combinations: 60%–40%, 70%–30%, 80%–20%, and 90%–10%. To improve the accuracy, we performed a manual tuning of the six different models using the hyperparameters displayed in Table VI. We sequentially explored different hyperparameter value combinations from the given possible value ranges and tuned them until we obtained a state-of-the-art accuracy of at least 96%. We evaluated the performance of each ML model on different training and testing data distributions as explained in Section VII-B and found out that a 70%–30% data distribution gives better accuracy for all six ML methods.

2) Voting Classification: This process uses ensemble learning to collect predictive outputs from various models and generates an output that minimizes the error and the overfitting. There are in general two voting classifier approaches: soft voting based on probability and hard voting based on maximum votes. Since fake news detection is a binary classification problem, we use *hard voting* that predicts a target variable Y based on the maximum votes mode given by different models M_i to a class:

$$Y = \text{mode}\{M_1(X), M_2(X), \ldots, M_n(X)\}$$

where X is predictor or input variable.

V. WELFAKE FAKE NEWS DETECTION ALGORITHM

Fig. 3 shows the WELFake binary news classification method (as real or fake) using TF-IDF and CV, three linguistic feature sets (LFS1, LFS2 and LFS3) and the hard voting classifier. The WELFake workflow consists of four steps:

1) Apply TF-IDF and CV on entire data set, store the results in P_1 and P_2, and decide the better WE based on accuracy.

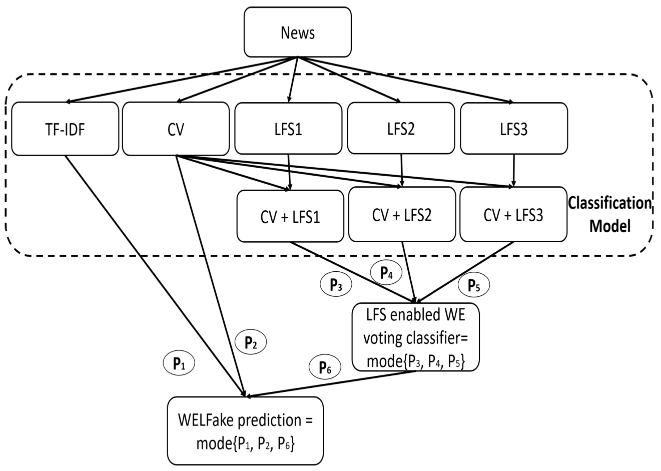

News
TF-IDF
CV
LFS1
LFS2
LFS3
CV + LFS1
CV + LFS2
CV + LFS3
Classification Model
P₃
P₄
P₅
LFS enabled WE voting classifier= mode{P₃, P₄, P₅}
P₁
P₂
P₆
WELFake prediction = mode{P₁, P₂, P₆}

Fig. 3. Sequential workflow of WELFake model.

2) Apply CV (better performing method) on three well-defined LFS and store the results in P_3, P_4, and P_5.
3) Apply WE hard voting classifier on P_3, P_4, and P_5, and generate the prediction P_6 as output.
4) Combine P_1, P_2, and P_6 using the hard voting classifier and generate the final prediction output.

Algorithm 1 explains the steps of the proposed WELFake model organized in four phases, as explained in Fig. 2.

1) *Data Set Preparation:* This the WELFake data collection in line 1 and data set preprocessing in line 2.
2) *Feature Engineering:* Extracts linguistic features of the data set in line 3 and applies Pearson's coefficient to select the significant linguistic features in line 4. Line 5 creates an odd number of LFS to perform voting.
3) *WE:* (i.e., CV and TF-IDF) This applies on the entire data set in lines 6 and 7. We select in line 8 the best method to combine with the various LFS created in line 5. Lines 9 and 10 combine the LFS with the best WE method.
4) *ML Model Tuning and Voting Classifier:* We train in line 11 the data sets from line 5 on various ML classification models and select the best results from each set. Line 12 applies the voting classifier on the results achieved on the different LFS using the best ML classification model and generates the hard voting output. Line 13 applies again the hard voting classifier (line 12), CV (line 6), and TF-IDF (line 7) and returns the final news classification prediction.

VI. CNN AND BERT IMPLEMENTATIONS ON WELFAKE DATA SET

This section describes the CNN and BERT implementation on the WELFake data set. We performed a hyperparameter tuning to improve their performance, summarized in Table VII.

A. CNN-Based Fake News Detection

After preprocessing the text as presented in Section IV-A2, we pass it to our CNN implementation consisting of several layers shown in Fig. 4.

Algorithm 1: WELFake Fake News Detection Algorithm

```
   Data: Kaggle, McIntire, Reuters, BuzzFeed
   Result: WELFake Model for news classification
   // Phase 1: data set preparation
1  WELFake_data set ← collection(Kaggle, McIntire, Reuters, BuzzFeed) // News collection
2  WELFake_data set ← preprocess(WELFake_data set) // Data set pre_processing
   // Phase 2: feature engineering
3  $LF$ ← extract(WELFake_data set) // Linguistic feature extraction
4  $LF$ ← selection(LF) // Feature selection using Pearson's correlation
5  $LFS$ ← split(LF) // Split linguistic features in odd sets
   // Phase 3: Word embedding
6  $CV$ ← cv(WELFake_data set) // Apply CV technique on data set
7  $TFIDF$ ← tfidf(WELFake_data set) // Apply TFIDF technique on data set
8  $Best$ ← select(CV, TFIDF) // Select best embedding technique with data set
9  foreach $LFS_i \in LFS$ do
10     $CLFS_i$ ← combine($CLFS_i$, Best)
   // Phase 4: ML model tuning and voting classifier
11 $Model$ ← bestModel(SVM (CLFS), DT (CLFS), NB (CLFS), Bagging (CLFS), AdaBoost (CLFS), KNN (CLFS))
    // Selection of best model for linguistic feature set
12 $vote\_hard$ ← votingClassifier(Model ($CLFS_i$));
13 return votingClassifier(TFIDF, CV, vote_hard)
```

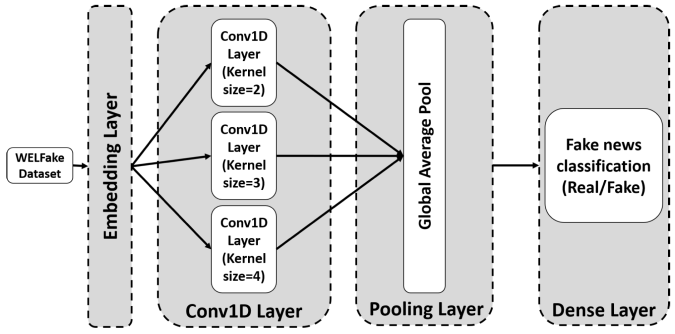

WELFake Dataset
Embedding Layer
Conv1D Layer (Kernel size=2)
Conv1D Layer (Kernel size=3)
Conv1D Layer (Kernel size=4)
Conv1D Layer
Global Average Pool
Pooling Layer
Fake news classification (Real/Fake)
Dense Layer

Fig. 4. CNN text processing on WELFake data set.

1) Embedding Layer: This converts the input text into WE, where every word in the vocabulary transforms into a high-dimensional space vector for next layer execution. We used the GloVe technique for the conversion of text into a vector.

2) Convolutional Layer: This layer the WE from the previous layer as input and specifies the number of Conv1D layers and the kernel size k, representing the number of words in a sentence during one filter slide. We used three Conv1D layers having kernels of size two, three, and four with 32 filters each (totalling 96 filters in the Conv1D layers). We obtained 32 feature maps from each kernel for further execution and calculated the filter height as $m - k + 1$, where m is the maximum sequence length in the data set.

3) Pooling Layer: The pooling layer reduces the number of outputs generated at the previous layer. After applying a global average pooling, the size of the vector is the same as the number of filters. This layer also merges the converted vectors of size $f \times x$, where f is the number of filters and x is the number of kernels.

4) Dense Layer: This layer takes the output from the pooling layer and passes it on to two consecutive dense layers for final prediction using a Sigmoid activation function with values in the $[0, 1]$ interval, which helps to assess the CNN prediction confidence.

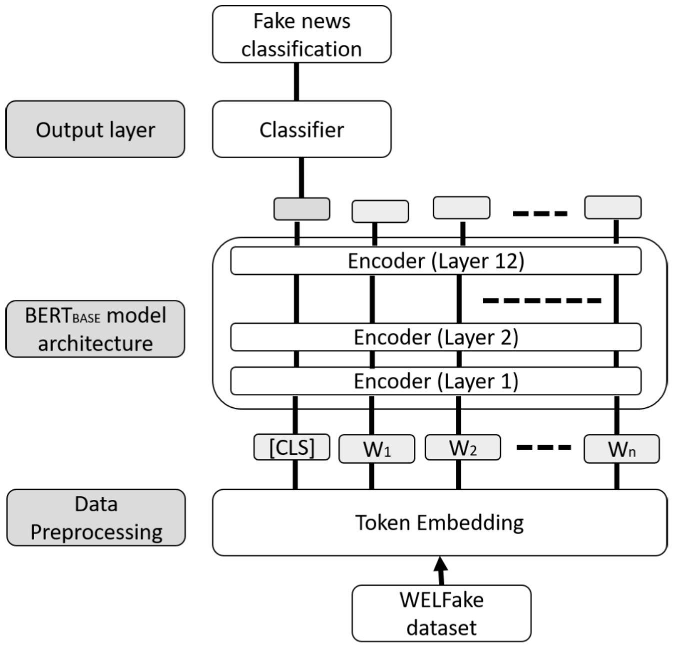


Fig. 5. BERT text processing on WELFake data set.

B. BERT-Based Fake News Detection

We performed fine-tuning of the pretrained BERT model for fake news classification in several steps, shown in Fig. 5.

1) Data Preprocessing: performs the operations described in Section IV-A2 on the raw text and adds a classification [CLS] token at the start of the first sentence.

TABLE VII
CNN AND BERT MODEL PARAMETERS

Model	*Parameter*	*Value*
CNN	Number of Conv1D layers	3
	Number of filters	32
	Kernel Size	2,3,4
	Number of global avgpool layer	3
	Activation function	Sigmoid function
	Optimizer	Adam
	Loss function	Binary-crossentropy
BERT	Number of dense layers	5
	Number of droupout layer	3
	Activation function	Sigmoid function

TABLE VIII
FAKE NEWS PREDICTION PARAMETERS

Evaluation parameter	*Predictive value*	*Actual value*
True positive (TP)	Yes	Yes
True negative (TN)	No	No
False positive (FP)	Yes	No
False negative (FN)	No	Yes

2) BERT Model Architecture: uses the $\text{BERT}_{\text{BASE}}$ model with 12 encoder layers. Each encoder layer performs a self-attention mechanism and passes the output to a feed-forward network. The output of each layer is a vector of default size 768. We applied at the 12th layer the [CLS] token to store the useful information for classification.

3) Output Layer: reads the vector stored in the [CLS] token. We used a Sigmoid activation function and implemented a NN with five dense layers and three dropout layers to classify the text news. The Sigmoid function produces the results in the range of [0, 1], which identifies confidence classes such that the class with a larger value wins the prediction.

VII. EXPERIMENTAL EVALUATION

We implemented WELFake in Python 3.6 on a Jupyter notebook. We run the experiments on a computer equipped with a ninth-generation i7 processor and 16 GB of memory.

A. Evaluation Metrics

We define four evaluation parameters, true-positive (TP), true-negative (TN), false-positive (FP), and false-negative (FN), based on the relation between the predictive news classification and the actual one, displayed in Table VIII. Based on these parameters, we evaluated the WELFake model on four performance metrics.

1) Accuracy: This is the ratio between the number of correct predictions and the total number of predictions

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}.$$

2) Precision: This measures the positive predicted value, as the ratio between the number of correct positive predictions to the total number of positive predictions

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}.$$

3) Recall: R measures the sensitivity of the model as the ratio between the number of correct positive predictions to the total number of correctly predicted results

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

4) F1-Score: $F1$ measures the testing accuracy of the model as the harmonic mean of the precision and the recall

$$F1 - \text{score} = \frac{2}{\text{Recall}^{-1} + \text{Precision}^{-1}}.$$

B. Comparative ML Classification

We analyze the accuracy of the WELFake model on the six ML classifiers in Table IX using four scenarios.

1) LFS: This classifies the news on the three LFS individually using the six ML classifiers. The accuracy of all three sets ranges between 77.3% and 85.6%. LFS1 gives the best accuracy of 83.4% using Bagging and the worst accuracy of 79.2% using NB. LFS2 produces the highest accuracy of 84.2% using Bagging and the lowest accuracy of 77.3% using NB. Similarly, LFS3 achieves the highest 85.6% accuracy using SVM and the lowest accuracy of 79.8% using NB. Bagging and SVM performed the best among all six ML classifiers followed by AdaBoost, DT, KNN, and NB.

2) WE: This applies CV and TF-IDF on the WELFake data set and classifies the news to predict P_1 and P_2, as illustrated in Fig. 3. We observed that CV achieved in general a better accuracy than TF-IDF. CV performed the best on SVM with a 95.61% accuracy, followed by Bagging, AdaBoost, NB, DT, and KNN. TF-IDF achieved an accuracy of 95.12% using Bagging, followed by SVM, AdaBoost, NB, KNN, and DT.

3) LFS-Enabled WE: This combines CV with the three LFS to predict P_3, P_4, and P_5 and applies voting classifier to obtain P_6. We achieved a maximum accuracy of up to 96.1% using SVM and a minimum accuracy of up to 89.6% using DT. Overall, SVM performed the best followed by Bagging, AdaBoost, NB, KNN, and DT.

4) WELFake Prediction: This predicts the final classification output by applying voting classifier across the LFS-enabled WE classification (P_6), TF-IDF (P_1) and CV (P_2) predictions. SVM achieved the maximum accuracy of 96.73% followed by AdaBoost, Bagging, NB, KNN, and DT.

5) Comparison: Table X shows the performance among several training and testing dataset splits and Table XI compares the overall performance of the WELFake model in terms of accuracy, precision, recall, and F1-score, introduced in Section VII-A. From all six ML models, WELFake produces the best results using SVM with a maximum accuracy of 96.73%, and a minimum accuracy of 89.92% using DT. Similarly, SVM achieved the highest precision (94.6%), recall (98.61%), and F-1 score (96.56%), while DT scored the lowest precision (86.1%) and F1-score (89.24%). KNN achieved the worst recall of 90.55% only. From these results, we conclude that the proposed WELFake model achieved the best performance on the WELFake data set (on all evaluation metrics) using SVM followed by AdaBoost, Bagging, NB, KNN, and DT.

TABLE IX
ACCURACY ANALYSIS OF WELFAKE MODEL

Scenarios $\xrightarrow{}$	*LFS*			*WE*		*LFSWE*			Final prediction
↓ *Model*	*LFS1 (%)*	*LFS2 (%)*	*LFS3 (%)*	*TF-IDF (%)*	*CV (%)*	*CV + LFS1 (%)*	*CV + LFS2 (%)*	*CV + LFS3 (%)*	Voting classifier (%)
KNN	79.6	81.8	80.6	89.6	88.2	90.3	90.5	90.1	90.16
SVM	82.5	83.5	85.6	94.5	95.61	95.6	96.1	95.01	**96.73**
NB	79.2	77.3	79.8	91.02	91.03	91.05	91.08	92.01	92.12
DT	81.4	79.6	80.8	89.54	89.51	90.1	89.61	89.68	89.92
Bagging	83.4	84.2	84.2	95.12	95.04	95.08	95.3	95.3	95.31
Adaboost	81.8	81.9	80.6	93.78	94.9	95.18	95.18	95.2	95.32

TABLE X
WELFAKE ACCURACY ANALYSIS ON DIFFERENT TRAINING AND TESTING DATA SETS

Training-testing	*Model*					
data ratios (%)	*KNN*	*SVM*	*NB*	*DT*	*Bagging*	*Adaboost*
60 – 40	89.64	95.23	92.01	88.97	95.30	95.12
70 – 30	**90.16**	**96.73**	**92.12**	**89.92**	**95.31**	**95.32**
80 – 20	90.16	96.71	92.12	89.91	95.32	95.30
90 – 10	90.15	96.73	92.12	89.92	95.30	95.30

TABLE XI
EVALUATION RESULTS FOR DIFFERENT ML MODELS

Model	*Accuracy (%)*	*Precision (%)*	*Recall (%)*	*F1-Score (%)*
KNN	90.16	89.02	90.55	89.78
SVM	96.73	94.60	98.61	96.56
NB	92.12	91.45	92.25	91.85
DT	89.92	86.10	92.62	89.24
Bagging	95.31	91.78	98.46	95.00
Adaboost	95.32	91.81	98.46	95.02

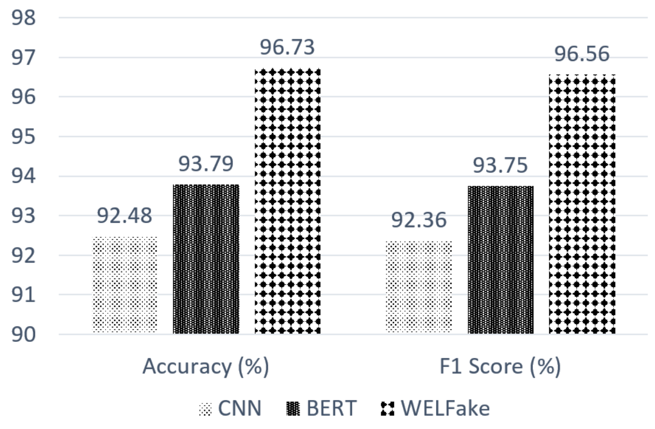


Fig. 6. CNN, BERT, and WELFake performance comparison.

TABLE XII
WELFAKE GENERALIZATION ACCURACY

No.	*Training dataset*		*Testing dataset*		*Accuracy*
1	BuzzFeed, Reuters, McIntire	71%	Kaggle	29%	96.70%
2	BuzzFeed, Reuters, Kaggle	91%	McIntire	9%	96.43%
3	BuzzFeed, McIntire, Kaggle	38%	Reuters	62%	96.72%
4	Reuters, McIntire, Kaggle	99%	BuzzFeed	1%	96.03%

C. Comparative Text Classification

Fig. 6 compares the accuracy and F1-score of the WELFake model with the CNN and BERT state-of-the-art methods. The WELFake model achieved a 96.73% accuracy, while CNN and BERT achieved a maximum accuracy only up to 92.48% and 93.79%, respectively. Similarly, WELFake also shows a better F1-score compared to CNN and BERT due to its better generalization. While Kaliyar *et al.* [36] achieved a 98.36% accuracy using a deep NN on single data set, its accuracy reduced to 92.48% on the WELFake data set. Similarly, BERT is a pretrained model which works well with labeled data, while its performance gets compromised in a generalized data set where testing data are independent of the training data. Finally, WELFake focuses on text writing pattern linguistic features that extract the text structure and provide the syntax, sentiment, grammatical, and readability evidence specific to the text content, which explains the improved fake news detection. Overall, WELFake achieved the highest accuracy followed by BERT and CNN.

D. WELFake Generalization

We use different training and testing data sets to analyze the generalization performance of the WELFake model. For this purpose, we followed an adversarial approach that splits the WELFake data set into four constituent subsets (i.e., BuzzFeed, Reuters, McIntire, and Kaggle). We generated four experimental data sets that combine three of them as training data and keeps the fourth for testing, as presented in Table XII.

1) Under-Fitting Data Set (3): This has the training data set smaller than the testing data set, which overestimates the results in over-constrained models.

2) Over-Fitting Data Set (4): This uses almost the entire data set as part of training, which underestimates the results in under-constrained models.

3) Balanced Data Set (1, 2): This splits the training and testing data in acceptable ratios.

The evaluation results reveal that the WELFake model accuracy remains high in all four scenarios, ranging between 96.03% to 96.84% with a very small standard deviation of 0.3. It is possible because our proposed WELFake model handles well the overfitting and underfitting data due to the multilevel model training and data classification. These results, therefore, validate the generalization of the proposed WELFake model.

E. Related Work Comparison

We compare our proposed WELFake model with three related models, summarized in Table XIII.

TABLE XIII
WELFAKE COMPARISON WITH RELATED METHODS

Parameter	*Ahmed* et al. *[48]*	*Shu* et al. *[49]*	*Gravanis* et al. *[37]*	*WELFake*
Dataset	1. Kaggle	1. Politifact 2. BuzzFeed	1. Kaggle-EXT 2. BuzzFeed 3. Politifact 4. McIntire 5. UNBiased	1. Kaggle 2. BuzzFeed 3. Reuters 4 McIntire 5 WELFake
Number of news articles	1. 25,200	1. 240 2. 182	1. 23,340 2. 240 3. 182 4. 6,310 5. 3,404	1. 20,800 2. 101 3. 44,898 4. 6,335 5. 72,134
Linguistic features	No	Yes	Yes	Yes
WE	TF-IDF	No	Word2Vec	CV
Other features	No	No	NA	Voting classifier
Number of features	1	2	57	20
Classifier	Linear SVM	SVM	SVM	SVM
Accuracy	1. 92%	1. 87.8% 2. 86.4%	1. 99.0% 2. 72.7% 3. 84.7% 4. 81.0% 5. 95.0%	1. 92.60% 2. 82.78% 3. 92.04% 4. 91.78% 5. 96.73%

1) Ahmed et al. [48]: experimented with fake news detection on the Kaggle-EXT data set with 25 200 articles. They did not use linguistic features and applied a linear SVM model on TF-IDF with the highest 92% accuracy.

2) Shu et al. [49]: used linguistic features for fake news detection on the BuzzFeed and Politifact data sets of only 240 and 182 articles. They separately used a linguistic feature method with two features and implemented SVM on both data sets with an accuracy of 87.8% for Politifact and 86.4% for BuzzFeed.

3) Gravanis et al. [37]: used the UNBiased data set with 3404 articles with 2004 real news and 1400 fake ones and achieved accuracy up to 95% using the SVM classifier. They used 57 linguistic features with the Word2vec WE method. They also compared their method on other data sets (i.e., Kaggle-EXT, BuzzFeed, Politifact, and McIntire) and achieved an accuracy of up to 99.0%, 72.70%, 84.7%, and 81%, respectively. They claimed of using a biased Kaggle-EXT data set (in the year 2018) with real news from one source only, which produced a constant performance regardless of the number of features (i.e., up to 99% accuracy even by using a single linguistic feature, such as a typo).

4) WELFake: applied on a larger data set with over 72 000 news achieved a higher accuracy of 96.73% compared to the related methods. For a fair comparison, we separately applied the WELFake model on the four smaller data sets (i.e., Kaggle, McIntire, Reuters, Buzzfeed) too. Compared to [37] (the better-proven method among [37], [48], [49]) on the Buzzfeed and McIntire data sets, WELFake improved the accuracy from 72.7% to 82.7%, respectively, from 81% to 91.78%.

VIII. CONCLUSIONS AND FUTURE WORK

We presented a new model called WELFake for text fake news detection. For this purpose, we prepared a larger data set called WELFake with over 72 000 news articles combining four open-source data sets (i.e., Kaggle, McIntire, Reuters, and BuzzFeed) to reduce their individual limitation and bias. Afterward, we analyzed over 80 linguistic features from state-of-the-art works and selected 20 significant ones to minimize the computational complexity and increase the standard classifiers' accuracy. We applied two WE-based methods (i.e., TF-IDF, CV) over these linguistic features using six ML models (i.e., KNN, SVM, NB, DT, Bagging, and AdaBoost) and found out that CV produces better overall accuracy than TF-IDF with an SVM model. We, therefore, used CV over LFS and classified the 20 features based on four categories: writing pattern, readability index, psycho-linguistics, and quantity.

As the number of predictors that participate in the voting classifier needs to be odd, we prepared three LFS by distributing the twenty selected features in a balanced manner across these categories. Afterward, we embedded CV with these LFS and applied all six ML models. We determined the most accurate ML model and took its predicted results from each WE-enabled LFS data set for voting classification. We finally applied the result of this voting classifier to the next level voting classification with the best model results of TF-IDF and CV over LFS and obtained the final classification. Experimental results show that the WELFake model produces a high 96.73% accuracy on the WELFake data set. To further analyze its advantage we compared it with two state-of-the-art works and found out that it improves the overall accuracy by 1.31% compared to BERT and 4.25% compared to CNN models. The proposed WELFake model also improved the accuracy by up to 10% on the McIntire and BuzzFeed data sets [37]. We also analyzed the performance of different ML models in terms of accuracy, precision, recall, and F1-score, and found out that SVM produced the most accurate results. Finally, our frequency-based model focused on analyzing writing patterns outperformed predictive-based related works implemented using the Word2vec WE method by up to 1.73%.

We plan to extend our work in the future with other factors like knowledge graphs and user credibility for further verification of the output generated by the WELFake model.

REFERENCES

[1] W. Jiang, J. Wu, F. Li, G. Wang, and H. Zheng, "Trust evaluation in online social networks using generalized network flow," *IEEE Trans. Comput.*, vol. 65, no. 3, pp. 952–963, Mar. 2016.
[2] M. Alrubaian, M. Al-Qurishi, A. Alamri, M. Al-Rakhami, M. M. Hassan, and G. Fortino, "Credibility in online social networks: A survey," *IEEE Access*, vol. 7, pp. 2828–2855, 2019.
[3] S. Ranganath, S. Wang, X. Hu, J. Tang, and H. Liu, "Facilitating time critical information seeking in social media," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 10, pp. 2197–2209, Oct. 2017.
[4] Z. Zhang, R. Sun, X. Wang, and C. Zhao, "A situational analytic method for user behavior pattern in multimedia social networks," *IEEE Trans. Big Data*, vol. 5, no. 4, pp. 520–528, Dec. 2019.
[5] M. Schudson and B. Zelizer, "Fake news in context," in *Understanding and Addressing the Disinformation Ecosystem*. Philadelphia, PA, USA: Annenberg School for Communication, Apr. 2017, pp. 1–4.
[6] S. Zaryan, "Truth and trust: How audiences are making sense of fake news," M.S. thesis, Media Commun. Studies, Lund Univ. Publications Student Papers, Stockholm, Sweden, Jun. 2017. [Online]. Available: https://lup.lub.lu.se/student-papers/search/publication/8906886
[7] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, Mar. 2018.
[8] R. Sequeira, A. Gayen, N. Ganguly, S. K. Dandapat, and J. Chandra, "A large-scale study of the Twitter follower network to characterize the spread of prescription drug abuse tweets," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1232–1244, Dec. 2019.
[9] *Politifact News Dataset*. Accessed: Mar. 31, 2020. [Online]. Available: http://www.politifact.com/
[10] *The Washingtonpost Fact Checker*. Accessed: Mar. 31, 2020. [Online]. Available: https://www.washingtonpost.com/news/fact-checker

[11] *Fact Check*. Accessed: Mar. 31, 2020. [Online]. Available: https://www.factcheck.org/

[12] *Snopes*. Accessed: Mar. 31, 2020. [Online]. Available: https://www.snopes.com/

[13] *Truthorfiction*. Accessed: Mar. 31, 2020. [Online]. Available: https://www.truthorfiction.com/

[14] *Fullfact*. Accessed: Mar. 31, 2020. [Online]. Available: https://fullfact.org/

[15] *Hoax Slayer*. Accessed: Mar. 31, 2020. [Online]. Available: http://hoax-slayer.com/

[16] *Viswas News*. Accessed: Mar. 31, 2020. [Online]. Available: http://www.vishvasnews.com/

[17] *Factly*. Accessed: Mar. 31, 2020. [Online]. Available: https://factly.in/

[18] L.-L. Shi *et al.*, "Human-centric cyber social computing model for hot-event detection and propagation," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 5, pp. 1042–1050, Oct. 2019.

[19] M. Glenski, T. Weninger, and S. Volkova, "Propagation from deceptive news sources who shares, how much, how evenly, and how quickly?" *IEEE Trans. Comput. Social Syst.*, vol. 5, no. 4, pp. 1071–1082, Dec. 2018.

[20] E. Lancaster, T. Chakraborty, and V. S. Subrahmanian, "$MALT^P$: Parallel prediction of malicious tweets," *IEEE Trans. Comput. Social Syst.*, vol. 5, no. 4, pp. 1096–1108, Dec. 2018.

[21] P. K. Verma and P. Agrawal, "Study and detection of fake news: P^2C^2-based machine learning approach," in *Proc. Int. Conf. Data Manage., Anal. Innov.*, vol. 1175. Singapore: Springer, Sep. 2020, pp. 261–278.

[22] Z. Jin, J. Cao, Y. Zhang, J. Zhou, and Q. Tian, "Novel visual and statistical image features for microblogs news verification," *IEEE Trans. Multimedia*, vol. 19, no. 3, pp. 598–608, Mar. 2017.

[23] B. Ratner, "The correlation coefficient: Its values range between $+1/-1$, or do they?" *J. Targeting, Meas. Anal. Marketing*, vol. 17, no. 2, pp. 139–142, Jun. 2009.

[24] A. De Salve, P. Mori, B. Guidi, and L. Ricci, "An analysis of the internal organization of Facebook groups," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1245–1256, Dec. 2019.

[25] P. K. Verma, P. Agrawal, and R. Prodan, *WELFake Dataset for Fake News Detection in Text Data (Version: 0.1) [Data Set]*. Genéve, Switzerland: Zenodo, 2021.

[26] V. Madaan and A. Goyal, "Predicting ayurveda-based constituent balancing in human body using machine learning methods," *IEEE Access*, vol. 8, pp. 65060–65070, 2020.

[27] M. Li, G. Clinton, Y. Miao, and F. Gao, "Short text classification via knowledge powered attention with similarity matrix based CNN," 2020, *arXiv:2002.03350*. [Online]. Available: http://arxiv.org/abs/2002.03350

[28] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune bert for text classification," in *Proc. China Nat. Conf. Chin. Comput. Linguistics*, vol. 11856. Cham, Switzerland: Springer, Feb. 2020, pp. 194–206. [Online]. Available: https://arxiv.org/abs/1905.05583

[29] K. Dzmitry Bahdanau, Y. Cho, and Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015, pp. 1–15. [Online]. Available: https://arxiv.org/abs/1409.0473

[30] Y. Chen, N. J. Conroy, and V. L. Rubin, "Misleading online content: Recognizing clickbait as 'false news,'" in *Proc. ACM Workshop Multimodal Deception Detection*, Nov. 2015, pp. 15–19.

[31] P. Bourgonje, J. Moreno Schneider, and G. Rehm, "From clickbait to fake news detection: An approach based on detecting the stance of headlines to articles," in *Proc. EMNLP Workshop, Natural Lang. Process. Meets Journalism*, 2017, pp. 84–89.

[32] H. Rashkin, E. Choi, J. Y. Jang, S. Volkova, and Y. Choi, "Truth of varying shades: Analyzing language in fake news and political fact-checking," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2931–2937.

[33] M. Alrubaian, M. Al-Qurishi, M. Mehedi Hassan, and A. Alamri, "A credibility analysis system for assessing information on Twitter," *IEEE Trans. Depend. Sec. Comput.*, vol. 15, no. 4, pp. 661–674, Aug. 2018.

[34] C. Castillo, M. Mendoza, and B. Poblete, "Information credibility on Twitter," in *Proc. 20th Int. Conf. World Wide Web (WWW)*, 2011, pp. 675–684.

[35] D. Benjamin, D. Horne, and S. Adali, "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 *Proc. 2nd Int. Workshop News Public Opinion*, Mar. 2017, pp. 1–9.

[36] R. K. Kaliyar, A. Goswami, P. Narang, and S. Sinha, "FNDNet—A deep convolutional neural network for fake news detection," *Cognit. Syst. Res.*, vol. 61, pp. 32–44, Jun. 2020.

[37] G. Gravanis, A. Vakali, K. Diamantaras, and P. Karadais, "Behind the cues: A benchmarking study for fake news detection," *Expert Syst. Appl.*, vol. 128, pp. 201–213, Aug. 2019.

[38] K. Shu, L. Cui, S. Wang, D. Lee, and H. Liu, "dEFEND: Explainable fake news detection," in *Proc. KDD 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Jul. 2019, pp. 395–405.

[39] R. Zellers *et al.*, "Defending against neural fake news," 2019, *arXiv:1905.12616*. [Online]. Available: http://arxiv.org/abs/1905.12616

[40] J. K. Burgoon, J. Blair, T. Qin, and J. Nunamaker, "Detecting deception through linguistic analysis," in *Proc. 1st NSF/NIJ Conf. Intell. Secur. Inform.*, Berlin, Germany: Springer, May 2003, pp. 91–101.

[41] M. D. Vicario, W. Quattrociocchi, A. Scala, and F. Zollo, "Polarization and fake news: Early warning of potential misinformation targets," *ACM Trans. Web*, vol. 13, no. 2, pp. 1–22, Apr. 2019.

[42] V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, "Automatic detection of fake news," in *Proc. 27th Int. Conf. Comput. Linguistics*, Santa Fe, NM, USA: Association for Computational Linguistics, Aug. 2018, pp. 3391–3401.

[43] C. Buntain and J. Golbeck, "Automatically identifying fake news in popular Twitter threads," in *Proc. IEEE Int. Conf. Smart Cloud (SmartCloud)*, New York, NY, USA, USA, Nov. 2017, pp. 208–215.

[44] T. Mitra and E. Gilbert, "Credbank: A large-scale social media corpus with associated credibility annotations," in *Proc. 9th Int. AAAI Conf. Web Social Media*. Apr. 2015, pp. 258–267.

[45] A. Zubiaga, G. W. S. Hoi, M. Liakata, and R. Procter, "PHEME dataset of rumours and non-rumours," Univ. Warwick, Coventry, U.K., Oct. 2016. [Online]. Available: https://figshare.com/articles/dataset/PHEME_dataset_of_rumours_and_non-rumours/4010619

[46] M. L. Newman, J. W. Pennebaker, D. S. Berry, and J. M. Richards, "Lying words: Predicting deception from linguistic styles," *Personality Social Psychol. Bull.*, vol. 29, no. 5, pp. 665–675, May 2003.

[47] L. Zhou, J. K. Burgoon, J. F. Nunamaker, and D. Twitchell, "Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated communications," *Group Decis. Negotiation*, vol. 13, no. 1, pp. 81–106, Jan. 2004.

[48] H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using N-gram analysis and machine learning techniques," in *Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, vol. 10618. Cham, Switzerland: Springer, Oct. 2017, pp. 127–138.

[49] K. Shu, S. Wang, and H. Liu, "Exploiting Tri-relationship for fake news detection," Dec. 2018, *arXiv:1712.07709v1*. [Online]. Available: https://arxiv.org/abs/1712.07709v1

[50] C. Burfoot and T. Baldwin, "Automatic satire detection: Are you having a laugh?" in *Proc. ACL-IJCNLP Conf. Short Papers ACL-IJCNLP*, 2009, pp. 161–164.

[51] B. Riedel, I. Augenstein, G. P. Spithourakis, and S. Riedel, "A simple but tough-to-beat baseline for the fake news challenge stance detection task," 2017, *arXiv:1707.03264*. [Online]. Available: http://arxiv.org/abs/1707.03264

[52] W. Y. Wang, "'Liar, liar pants on fire': A new benchmark dataset for fake news detection," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics (Short Papers)*, vol. 2, 2017, pp. 422–426.

[53] *Mcintire Fake News Dataset*. Accessed: Apr. 15, 2020. [Online]. Available: https://github.com/lutzhamel/fake-news

[54] *Fake News Kaggle Dataset*. Accessed: Apr. 15, 2020. [Online]. Available: https://www.kaggle.com/c/fake-news/data?select=train.csv

[55] *Benjamin Political News Dataset*. Accessed: May 15, 2020. [Online]. Available: https://github.com/rpitrust/fakenewsdata1

[56] *Burfoot Satire News Dataset*. Accessed: May 15, 2020. [Online]. Available: http://www.csse.unimelb.edu.au/research/lt/ resources/satire

[57] *Buzzfeed News Dataset*. Accessed: May 15, 2020. [Online]. Available: https://github.com/BuzzFeedNews/2016-10-facebook-fact-check/tree/master/data

[58] *Credbank Dataset*. Accessed: May 15, 2020. [Online]. Available: http://compsocial.github.io/CREDBANK-data

[59] *Fake News Challenge Dataset*. Accessed: May 15, 2020. [Online]. Available: https://github.com/FakeNewsChallenge/fnc-1

[60] *Fakenewsnet Dataset*. Accessed: May 15, 2020. [Online]. Available: https://github.com/KaiDMML/FakeNewsNet

[61] *Liar Dataset*. Accessed: May 15, 2020. [Online]. Available: https://www.cs.ucsb.edu/~william/data/liar_dataset.zip

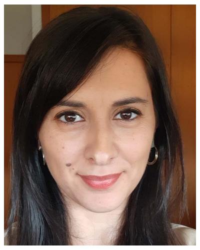
[62] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.
[63] G. Shmueli, N. R. Patel, and Bruce, *Data Mining for Business Intelligence: Concepts, Techniques, and Applications in Microsoft Office Excel With XLMiner*. Hoboken, NJ, USA: Wiley, 2007.
[64] *Introduction to TF-IDF*. Accessed: Jun. 15, 2020. [Online]. Available: http://www.tfidf.com/

Pawan Kumar Verma (Member, IEEE) received the M.Tech. degree in computer science from the Jaypee University of Information Technology, Waknaghat, India, in 2010. He is currently pursuing the Ph.D. degree with Lovely Professional University, Phagwara, India.

He has more than eight years of teaching and research experience in opinion mining, pattern recognition, image processing, and artificial intelligence. He has authored or coauthored more than eight research papers in various international conferences and journals of repute.

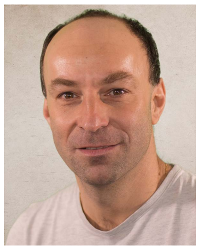
Prateek Agrawal received the Ph.D. degree from IKG-Punjab Technical University, Ajitgarh, India, in 2018.

He is currently a Post-Doctoral Researcher with the University of Klagenfurt, Klagenfurt, Austria, and an Associate Professor with the School of Computer Science Engineering, Lovely Professional University, Phagwara, India. He has authored or coauthored more than 60 research papers in various peer-reviewed journals and conferences. His research interests include natural language processing, machine learning, image processing, scheduling, and parallel processing.



Ivone Amorim received the Ph.D. degree in computer science from the University of Porto, Porto, Portugal, in 2016.

She currently is a Researcher with MOG Technologies, Moreira, Portugal. She is an Associated Member of the CMUP Mathematical Research Center, University of Porto. She participated in several national and international projects. She researched in the last years at several Portuguese institutions in the area of mathematics and its applications to several fields like robotics, power systems, and cryptography.



Radu Prodan (Member, IEEE) received the Ph.D. degree from the Vienna University of Technology, Vienna, Austria, in 2004.

He was an Associate Professor with the University of Innsbruck, Innsbruck, Austria, until 2018. He is currently Professor in distributed systems with the Institute of Software Technology, University of Klagenfurt, Klagenfurt, Austria. He is the Coordinator of the European Horizon 2020 project ARTICONF that researches a toolset for trustworthy, resilient, and globally sustainable decentralized applications, with a special focus on social media networks. He was involved in numerous national and European projects with a total budget of over six million. He has authored or coauthored more than 200 publications.

Dr. Prodan was a recipient of the IEEE best paper awards.