International Journal of Advanced Multidisciplinary Application | IJAMA
Volume 3 Issue 11 Nov 2026
ISSN No: 3048-9350

AI-Powered Fake News Detection Using Natural Language Processing

[1]Nakul Saraf, [2]Girdhari Lal
[1]Student, Department of Artificial Intelligence and Data Science
[2]Assistant Professor, Department of Artificial Intelligence and Data Science
[1,2]Poornima Institute of Engineering and Technology Jaipur, India
[1]nakulsaraf8716@gmail.com,[2]girdhari.lal@poornima.org ,

Abstract:
Automated detection of fake news, meaning fabricated news-style content that may circulate as misinformation or disinformation, is motivated by the speed and volume at which online content spreads and by the limited scalability of manual fact-checking. This paper treats fake-news detection as binary text classification and evaluates a hybrid model that fuses a fine-tuned BERT encoder, a convolutional branch operating over BERT token representations, and an auxiliary branch built from TF-IDF/SVD features and seven handcrafted linguistic features. The model is compared with Naive Bayes, logistic regression, a linear SVM, a word-embedding CNN, an LSTM and fine-tuned BERT on the WELFake dataset [1] (62,833 articles after cleaning; 55.36% real and 44.64% fake), using a stratified 80/10/10 split repeated with random seeds 42, 43 and 44. Over the three runs, the hybrid obtained mean test accuracy, precision, recall and F1-score of 99.62%, 99.64%, 99.51% and 99.58%, compared with 99.45%, 99.42%, 99.35% and 99.38% for fine-tuned BERT; its mean ROC-AUC (0.99968) was slightly lower than that of BERT (0.99984). No significance test was performed, and with three runs the differences should be interpreted cautiously. The principal limitation is that evaluation is restricted to a single benchmark, so robustness to unseen publishers, topics and time periods is untested. Ablation of the individual branches, cross-dataset evaluation, inference-latency measurement and prediction-level explanation are left for future work.
Keywords: Fake news detection, Natural language processing, BERT, Text classification, Deep learning, Misinformation, Linguistic features, WELFake.

1. INTRODUCTION

Online publishing and social networking have lowered the cost of producing and redistributing news, and false news stories have been shown to diffuse farther and faster than true ones on social media [2]. Exposure to fabricated stories has also been documented around the 2016 U.S. presidential election [3], [4]. Terminology in this area is not uniform [5], [6], [7]. In this paper, misinformation denotes false or misleading information shared without intent to deceive; disinformation denotes false information created or spread deliberately to deceive; and fake news denotes fabricated content that imitates the form of news reporting [5] and may circulate as either. Because the dataset used here records the veracity label of each article but not the intent behind it, the task studied is fake-news classification; the data do not allow misinformation to be separated from disinformation. Manual fact-checking is labour-intensive and difficult to scale to the volume of online content [6], [7], which motivates automated, natural language processing (NLP) based screening [8] that can support, but not replace, human verification.

Early automated approaches combined hand-crafted lexical and stylistic features with classical classifiers such as support vector machines (SVM) and Naive Bayes [9], [10], [11]. Later work applied convolutional (CNN) and recurrent (LSTM) networks [12], [13], [14], [15] and, more recently, pretrained transformers such as BERT [16], [17]. Several studies also combine BERT with convolutional or recurrent layers [18], [19], [20]. Whether such additions help relative to a fine-tuned BERT baseline and to strong linear baselines depends on the dataset, the split and the training protocol, which differ between studies. This paper therefore compares a range of model families under one protocol: Naive Bayes, logistic regression and a linear SVM as inexpensive classical references; a word-embedding CNN and an LSTM as neural baselines without transformer pretraining; fine-tuned BERT as the pretrained-transformer baseline; and a hybrid model whose added components are under test.

Combining BERT, convolutional layers and linguistic features is not claimed to be new, because related hybrid architectures have been published [18], [19], [20], [21]. The contribution of this paper is limited to (i) a specified implementation of one such hybrid, in which a CNN operates over BERT token representations and is fused with a TF-IDF/SVD and linguistic-feature branch; (ii) an empirical comparison with six established baselines on WELFake under identical stratified splits and three random seeds, reported with sample standard deviations and aggregated confusion-matrix counts; and (iii) an explicit statement

of what these experiments do not establish. Only WELFake was evaluated. Cross-dataset generalisation, ablation of the individual branches, inference latency and prediction-level explanation were neither evaluated nor implemented, and the system has not been deployed for content moderation. The remainder of the paper formulates the problem and objectives (Section 2), reviews related work (Section 3), describes the implemented framework (Section 4) and the experimental methodology (Section 5), reports and discusses the results (Section 6), and discusses limitations and ethical considerations (Section 7) before concluding (Section 8).

2. PROBLEM STATEMENT AND OBJECTIVES

2.1 Problem Formulation

Fake-news detection is formulated as a supervised binary text-classification task. Given a news article x, a function f(x) assigns a label $y \in \{0, 1\}$, where $y = 0$ denotes a real article and $y = 1$ denotes a fake article; Fake is the positive class. The input x is the article title concatenated with the article body. The model outputs a score $p = P(y = 1 \mid x) \in [0, 1]$, obtained by applying the sigmoid function to a single output logit, and predicts Fake if $p \geq 0.5$ and Real otherwise. A model confidence score $c = \max(p, 1 - p)$ is derived from the predicted class probability. This score has not been calibrated or evaluated for calibration and must not be interpreted as the probability that the prediction is correct.

2.2 Measurable Objectives

The following measurable objectives were defined for the experiments. The status of each objective after the experiments is stated with the objective.

- O1 — Evaluate the classification performance of the implemented hybrid model against the selected baseline models on the WELFake dataset using accuracy, precision, recall, F1-score and ROC-AUC. Status: completed; the hybrid obtained higher mean accuracy, precision, recall and F1-score than fine-tuned BERT but a lower mean ROC-AUC (Table II, Section 6).
- O2 — Evaluate the detection of fake news using fake-class recall and the false-negative rate, FNR = FN / (FN + TP), and compare these metrics with those of the baseline models. No target threshold was fixed before the experiments. Status: completed for the recorded test runs (Table II, Table III, Section 6).
- O3 — Provide prediction-level interpretability. Status: not met. The model wrapper returns the fake-class probability; the binary label and score-derived confidence can be calculated from it, but no influential-word or feature explanation is implemented. This objective is deferred to future work (Sections 4.5 and 8).
- O4 — Assess inference latency for use in moderation pipelines. Status: not met. No latency measurements were made; latency evaluation under a specified device and batch configuration remains future work (Sections 7.10 and 8).
- O5 — Assess robustness across at least two structurally different datasets. Status: not met. Only WELFake was evaluated, and cross-dataset evaluation remains future work (Sections 7.7, 7.11 and 8).

3. LITERATURE REVIEW

This section reviews five methodological families that are important for fake-news detection: (a) traditional machine learning with bag-of-words or TF-IDF features, (b) linguistic and stylometric features, (c) CNN and RNN sequence models, (d) transformer-based contextual models, including BERT-based hybrids, and (e) ensemble and voting models. Reported accuracies in the studies below were obtained on different datasets, splits and preprocessing and are therefore not directly comparable with one another or with the present results. Section 3.6 positions the present work.

3.1 Traditional Machine Learning Approaches

Classical classifiers such as Naive Bayes, logistic regression, SVM, decision trees and random forests, trained on TF-IDF or n-gram features, are standard baselines. Ahmed et al. [9] evaluated several classifiers on TF-IDF n-gram features and found that a linear SVM performed best, with accuracy of about 92%; the same group extended text-classification methods to opinion spam and fake news [22]. These methods are fast and comparatively transparent, but bag-of-words representations ignore word order and long-range context, and their results on one corpus may partly reflect vocabulary specific to that corpus.

3.2 Linguistic-Feature Methods

Pérez-Rosas et al. [10] introduced the FakeNewsAMT and Celebrity datasets and trained linear SVM classifiers on lexical, syntactic and readability-related features, reporting lower performance when training and test domains differed. Horne and Adali [11] compared stylistic, complexity and psychological features of real, fake and satirical articles and found systematic differences in titles and body text. Gravanis et al. [23] benchmarked content-based feature sets and classifiers, and Kwon et al. [24] examined linguistic, temporal and structural features of rumour propagation. Such features are interpretable, but the cross-domain results in [10] indicate that their discriminative value can depend on topic and domain.

3.3 CNN and RNN Architectures

Neural sequence models learn representations directly from text. Kim [12] showed that a CNN with parallel filter widths over pretrained word vectors performs well on sentence classification; that study did not address fake news but is the origin of the n-gram-style convolutional design used in later fake-news work. Ruchansky et al. [13] proposed CSI, which combines an LSTM-based text representation with user-behaviour information. Nasir et al. [14] combined CNN and LSTM layers on ISOT and FA-KES and reported better results than the single-architecture models they compared. Kaliyar et al. [15] proposed FNDNet, a deep CNN with GloVe embeddings evaluated on a Kaggle fake-news dataset. Dong et al. [25] used two CNN paths for semi-supervised learning from limited labelled data on LIAR [26] and PHEME [27]. These models rely on trained-from-scratch or static embeddings and therefore lack the pretrained contextual knowledge of transformers.

3.4 Transformer / BERT-Based Models

BERT [16] is pretrained on large unlabelled corpora (BooksCorpus and English Wikipedia) and fine-tuned for downstream tasks; the original study evaluated general language-understanding tasks, not fake news. FakeBERT [17] passes BERT embeddings to parallel convolutional blocks and reported 98.90% accuracy on a real-world fake-news dataset. MCred [18] concatenates a BERT branch with a CNN branch that uses kernel widths of two, three and four over GloVe embeddings; it was evaluated on four datasets, including WELFake, where it reported 99.01% accuracy. FakeStack [19] stacks BERT embeddings, a deep CNN with skip convolutions and an LSTM, and reported 98.25% accuracy on WELFake and 75.58% on LIAR, which illustrates the large dependence of results on the dataset. Al-Quayed et al. [20] combined BERT with bidirectional LSTM and GRU layers for news headlines in WELFake and compared the result with standard machine-learning, deep-learning and vanilla BERT models. Across these studies, the extent to which the added layers improve on a carefully fine-tuned BERT baseline is usually reported on a single split of one dataset.

3.5 Ensemble and Voting Methods

Ensemble methods combine several base learners through bagging, boosting or voting to reduce variance. Verma et al. [21] combined word-embedding representations with linguistic features in a multi-stage voting classifier, reported 96.73% accuracy, and reported gains over BERT and CNN models in their experiments. The same work introduced the WELFake dataset [1] by merging four earlier sources (Kaggle, McIntire, Reuters and BuzzFeed Political) to reduce dependence on any single collection; merging heterogeneous sources, however, does not by itself guarantee robustness to unseen publishers.

3.6 Positioning of the Present Work

Relative to MCred [18], in which the CNN uses GloVe embeddings in a branch parallel to BERT, the CNN in the present model operates on BERT's own contextual token representations, and a separate TF-IDF/SVD and linguistic-feature branch is added. Relative to the WELFake model [21], which classifies word-embedding and linguistic features through a voting ensemble, the statistical and linguistic features are here fused inside a single network that is trained jointly with BERT. These are architectural differences only. No novelty beyond the implementation and its empirical comparison is claimed, and no ablation has been performed to show that the convolutional or the auxiliary branch contributes to the observed performance.

TABLE I. COMPARISON OF REPRESENTATIVE PRIOR STUDIES ON FAKE NEWS DETECTION

Study	Approach / Model	Dataset(s)	Reported Focus
Nasir et al. [14]	Hybrid CNN-RNN (LSTM)	ISOT, FA-KES	Combines local (CNN) and sequential (LSTM) feature learning

Study	Approach / Model	Dataset(s)	Reported Focus
Dong et al. [25]	Two-path semi-supervised CNN	LIAR, PHEME	Learns from limited labeled + large unlabeled data
Verma et al. [21]	Linguistic features + word embedding voting (WELFake)	Kaggle, McIntire, Reuters, BuzzFeed	Merges 4 datasets into WELFake; combines linguistic features with word-embedding voting; reported 96.73% accuracy
Verma et al. [18]	BERT + CNN fusion (MCred)	WELFake, Kaggle, McIntire, FakeNews	Fuses global (BERT) and local (CNN over GloVe) semantics; reported 99.01% accuracy on WELFake
Ahmed et al. [9]	N-gram + TF-IDF + SVM/LR	ISOT	Classical ML baseline using n-gram analysis
Pérez-Rosas et al. [10]	Linguistic feature SVM	Celebrity, FakeNewsAMT	Cross-domain linguistic feature generalization
Kaliyar et al. [15]	FNDNet deep CNN + GloVe	Kaggle	Deep CNN with pretrained embeddings
Devlin et al. [16]	BERT pretraining (general NLP)	BooksCorpus, Wikipedia	Foundational bidirectional transformer model
Keya et al. [19]	BERT + deep CNN (skip convolutions) + LSTM (FakeStack)	LIAR, WELFake	Stacked hybrid; reported 98.25% accuracy on WELFake and 75.58% on LIAR
Al-Quayed et al. [20]	BERT + bidirectional LSTM and GRU layers	WELFake (news headlines)	Hybrid transformer model compared with standard ML/DL models and vanilla BERT

4. IMPLEMENTED HYBRID FRAMEWORK

This section describes the hybrid model as implemented and evaluated in this study, which is shown in Fig. 1. Only the components described below were implemented and evaluated; other components that would be needed in an operational system (multi-source ingestion, explanation, monitoring) are not part of the present implementation.

4.1 Dataset Ingestion and Cleaning

The implemented pipeline reads the WELFake CSV file [1], which contains an article index, a title, article text and a binary label. No other input format (for example JSON files, RSS feeds or database exports) is implemented, and the pipeline has been run only on WELFake; support for other sources would be a future extension. During loading, missing titles and text fields are replaced with empty strings and rows with missing labels are discarded. Articles whose body text contains fewer than 50 characters are removed, after which exact duplicates of the combined title-and-body string are removed. These operations reduce the 72,134 original records to 62,833 articles, so 9,301 records were removed. The removals are the combined effect of the cleaning operations and are not attributed to a single operation. Near-duplicate detection, HTML or URL removal and encoding repair are not performed. Labels are used as stored in the distributed file, with 0 denoting Real and 1 denoting Fake in this implementation. The cleaned set contains 34,786 Real articles (55.36%) and 28,047 Fake articles (44.64%).

4.2 Text Preprocessing

The article string used by every model is the title concatenated with the body text. Missing values, short articles and exact duplicates are handled during cleaning (Section 4.1). For the BERT branch, the bert-base-uncased tokenizer [16] performs lowercasing and WordPiece tokenisation, adds the [CLS] and [SEP] special tokens, and truncates or pads each sequence to 256 tokens; this lowercasing is a property of the tokenizer and applies to the BERT encoder and to the CNN that consumes its outputs. The auxiliary branch derives TF-IDF features and linguistic features from the article text as described in Section 4.3.

No lemmatisation is applied. The TF-IDF vectoriser uses the built-in English stop-word list; BERT inputs do not undergo a separate stop-word-removal step.

4.3 Feature Extraction

Three kinds of representation are used. First, BERT produces a contextual 768-dimensional vector for each of up to 256 tokens and a document-level vector at the [CLS] position. Second, a TF-IDF representation of the article text is reduced by truncated singular value decomposition (SVD) to 200 dimensions, giving a dense, low-dimensional representation of lexical statistics. The TF-IDF vectoriser uses at most 50,000 features, unigrams and bigrams, sublinear term-frequency scaling, a minimum document frequency of 3, and English stop-word removal. Third, seven handcrafted linguistic features are computed from the article text: log-transformed word count; exclamation-mark density per 100 characters; question-mark density per 100 characters; uppercase-character ratio; mean word length; lexical diversity (unique lowercased tokens divided by token count); and quotation-mark density per 100 characters. The 200 SVD values and seven linguistic features are concatenated and standardised using a StandardScaler fitted on the training partition. The auxiliary feed-forward network maps this vector through a 256-unit linear layer with ReLU activation and dropout of 0.3, followed by a 128-unit linear layer with ReLU activation. The TF-IDF vocabulary, SVD projection and scaler are fitted on training data only and then applied unchanged to validation and test data. Linguistic and stylistic cues have been studied in earlier work [10], [11]. The auxiliary representation is combined with the contextual representation because lexical and stylistic statistics may carry complementary information; whether the branches improve performance individually was not tested because no ablation experiment was run (Section 7.8).

4.4 Classification Model

The classifier has three branches whose outputs are fused (Fig. 1). Branch 1 (BERT): the tokenised article is encoded by bert-base-uncased (12 layers, 768 hidden units, 12 attention heads [16]), giving a matrix H of contextual token vectors (up to 256 × 768); the vector at the [CLS] position is used as the document-level representation h_cls. Branch 2 (CNN over BERT representations): one-dimensional convolutions with kernel widths of 2, 3 and 4 are applied over H, each with 128 filters and followed by max pooling over positions; the pooled outputs are concatenated into h_cnn. Parallel filters of several widths follow [12], [18], but here they operate on contextual BERT vectors instead of static word embeddings. Branch 3 (auxiliary features): the 200-dimensional TF-IDF/SVD vector and the seven standardised linguistic features pass through the auxiliary network (256-unit ReLU layer, dropout 0.3, then 128-unit ReLU layer) to give h_aux (Section 4.3).

Feature fusion and output: the three representations are concatenated, [h_cls; h_cnn; h_aux], and passed in order through layer normalisation, dropout with probability 0.1 and a final linear layer that produces a single logit z. With a 768-dimensional [CLS] vector, three 128-dimensional pooled CNN outputs and a 128-dimensional auxiliary representation, the fused vector has 1,280 dimensions. The fake-news probability is $p = \sigma(z) = 1 / (1 + e^{(-z)})$. The decision rule uses a threshold of 0.5: the article is classified as Fake if $p \geq 0.5$ and as Real if $p < 0.5$.The fine-tuned BERT baseline uses the same encoder with a classification head on the document-level representation, without the CNN and auxiliary branches. The classical, CNN and LSTM baselines are trained on the same partitions; their settings are listed in Section 5.4.

4.5 Prediction Output and Explainability

The prediction interface implemented by the model wrappers returns the fake-news probability p. The predicted binary label can be derived by applying the 0.5 threshold (Fake if $p \geq 0.5$, Real otherwise), and the score-derived confidence can be calculated as $c = \max(p, 1 - p)$; the wrapper does not package the label and confidence as separate return values. Neither the wrapper nor the evaluation pipeline returns influential words or feature-level explanations, and the implementation does not include SHAP, Integrated Gradients or attention-based explanations; no prediction examples with explanations are therefore presented. The probability p is the model's output score for the Fake class and is not guaranteed to be calibrated. The derived confidence c indicates the strength of the model's preference for its predicted class, not the probability that the prediction is correct. Feature attribution describes how the output depends on inputs; a causal explanation of whether an article is false requires evidence beyond the text. Attention weights alone do not establish why a model made a prediction. Post-hoc attribution methods such as SHAP or Integrated Gradients are possible future additions, but they would need to be implemented and validated first.

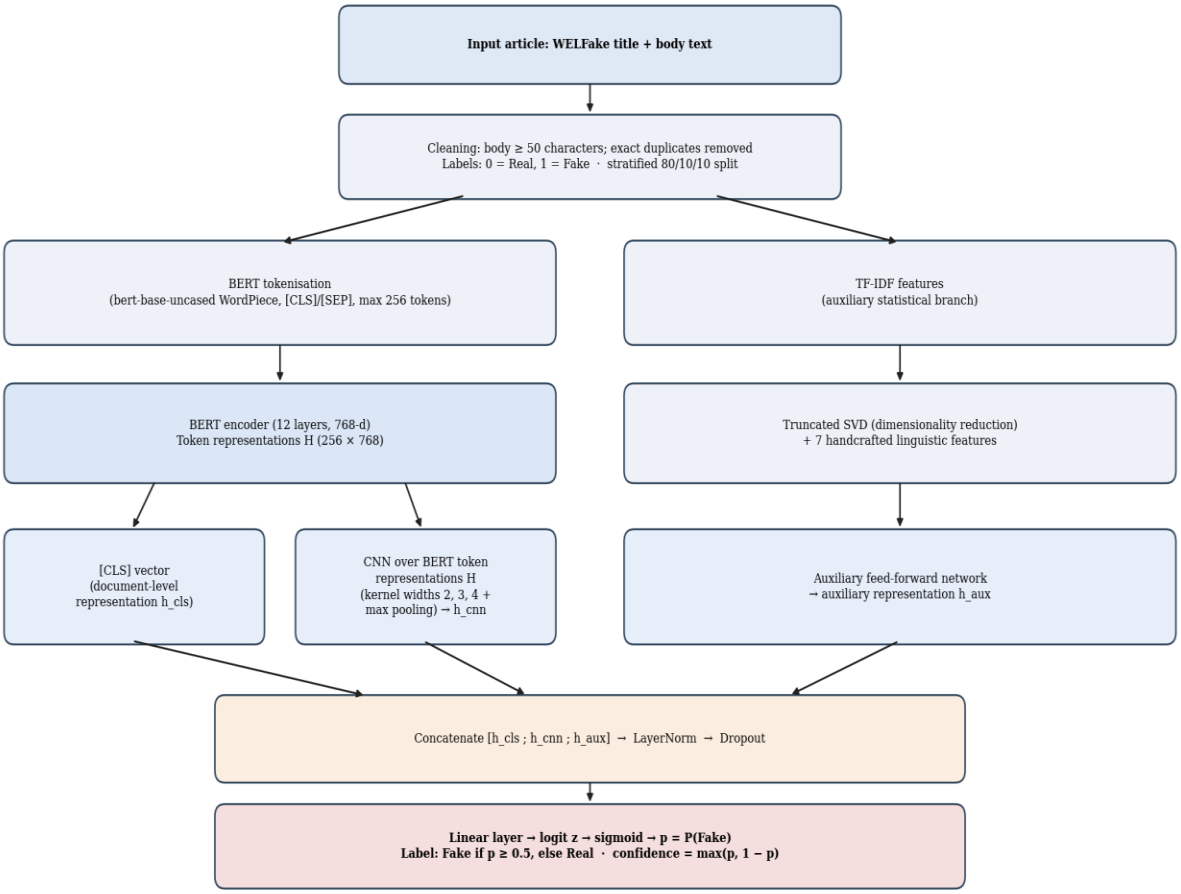


Fig. 1. Architecture of the implemented hybrid model: cleaned WELFake article (title + body) → BERT encoder with [CLS] vector and CNN over token representations, and TF-IDF/SVD with seven linguistic features and an auxiliary network → concatenation, layer normalisation, dropout and a linear output layer → fake-class probability p; the binary label and score-derived confidence are calculated from p. Prediction-level explanations are not implemented.

5. METHODOLOGY

5.1 Dataset Selection

For reproducibility, publicly documented benchmark datasets are preferable, for example ISOT [9], FA-KES [28], LIAR [26] and the Kaggle fake-news dataset [29]. A suitable dataset should have a documented labelling procedure and enough examples to fine-tune a transformer. This study uses only the WELFake dataset [1], [21], which merges four earlier sources and contains 72,134 records, of which 37,106 (51.44%) carry label 1 and 35,028 (48.56%) carry label 0. The labelling in the dataset description is internally inconsistent: it states 0 = fake and 1 = real, but also states 35,028 real and 37,106 fake articles, whereas label 0 occurs 35,028 times in the distributed file. The coding used here (0 = Real, 1 = Fake) follows the stated class counts and the description in [18]. After cleaning (Section 4.1), 62,833 articles remained: 34,786 Real (label 0; 55.36%) and 28,047 Fake (label 1; 44.64%). The cleaned dataset is therefore not balanced, and cleaning changed the class proportions: 9,059 articles of label 1 but only 242 of label 0 were removed. The reasons for each removal are not broken down here.

5.2 Class Distribution and Data Splits

The cleaned data were divided by stratified sampling into training, validation and test partitions in an 80/10/10 ratio. Stratification preserves the cleaned class proportions (55.36% Real and 44.64% Fake) in each partition, not the 51%/49%

proportions of the raw file. Each test partition contained 6,284 articles per run, of which 2,805 were Fake and 3,479 were Real, according to the recorded confusion-matrix counts. The experiment was repeated with random seeds 42, 43 and 44. The seed is passed as the random state to both stages of the stratified split, so the train, validation and test partitions may differ between runs. Within a given run, all seven models use the same partition. Because the test partitions across runs may overlap, the 18,852 evaluation instances aggregated over three runs (3 × 6,284) are not necessarily 18,852 distinct articles.

5.3 Preprocessing Pipeline Configuration

The preprocessing of Section 4.2 is applied consistently to the training, validation and test partitions. Short articles and exact duplicates are removed before the data are split, which prevents identical article strings from appearing in more than one partition. Near-duplicate articles, such as lightly edited copies of the same story, are not detected, so residual overlap cannot be excluded; data leakage is therefore reduced but not eliminated. The TF-IDF vectoriser is fitted on training text only using max_features = 50,000, ngram_range = (1, 2), sublinear_tf = True, min_df = 3 and stop_words = 'english'. The hybrid's truncated SVD projection uses 200 components and is fitted on training data only. The StandardScaler is fitted on the combined SVD and linguistic features from the training partition only. These fitted transformations are applied unchanged to validation and test data, avoiding use of held-out features during fitting. The BERT tokenizer is pretrained and has no fitted parameters.

5.4 Model Configuration

The BERT baseline and hybrid both use bert-base-uncased (12 transformer layers, 768 hidden dimensions and 12 attention heads [16]) with a maximum sequence length of 256 tokens. The fine-tuned BERT baseline uses the [CLS] representation, dropout of 0.1 and a single linear output layer. It is trained for up to three epochs with AdamW, learning rate 2×10^{-5}, weight decay 0.01 and batch size 16; early-stopping patience is one epoch. The hybrid uses 128 CNN filters for each kernel width (2, 3 and 4), max pooling, and an auxiliary feature branch consisting of TF-IDF, 200-component SVD and seven linguistic features. The auxiliary network contains a 256-unit ReLU layer, dropout of 0.3 and a 128-unit ReLU layer. The concatenated 1,280-dimensional representation is normalised with LayerNorm, passed through dropout of 0.1 and mapped to one logit by a final linear layer. The hybrid uses AdamW with differential learning rates of 2×10^{-5} for the pretrained BERT encoder and 1×10^{-3} for the newly initialised layers, weight decay 0.01, batch size 16, a maximum of five epochs and early-stopping patience of two epochs. The encoder is frozen for the first epoch and unfrozen at the start of epoch two. The BERT and hybrid learning-rate schedulers use linear decay with warm-up over 6% of the total training steps. The hybrid-specific settings are reported as the fixed settings used; the available records do not establish that a systematic hyperparameter search was performed.

The classical models share the training-fitted TF-IDF vectoriser described in Section 5.3. Their hyperparameters are selected using validation F1: Multinomial Naive Bayes tests $\alpha \in \{0.1, 0.5, 1.0\}$; logistic regression tests $C \in \{1, 10, 100\}$ with max_iter = 1,000; and the linear SVM tests $C \in \{0.1, 1, 10\}$. The CNN and LSTM baselines use a vocabulary of up to 30,000 tokens built from the training partition, lowercase regex tokenisation and a maximum sequence length of 400 tokens. Both use trainable 128-dimensional word embeddings. The CNN has 128 filters for each kernel width (2, 3 and 4), max-over-time pooling and dropout of 0.5. The LSTM is a single-layer model with 128 hidden units, masking of padded positions before max pooling and dropout of 0.5. Both neural baselines use AdamW with learning rate 1×10^{-3}, weight decay 0.01, batch size 64, a maximum of eight epochs and early-stopping patience of two, monitored by validation F1.

5.5 Training Procedure

The neural models are trained with binary cross-entropy with logits (BCEWithLogitsLoss); the loss is calculated from logits, and sigmoid is applied to obtain probabilities for validation and prediction. AdamW is used with weight decay 0.01, and the global gradient norm is clipped at 1.0 in the shared training loop. Fine-tuned BERT uses batch size 16, a maximum of three epochs and early-stopping patience of one. The hybrid uses batch size 16, a maximum of five epochs and patience of two. The CNN and LSTM baselines use batch size 64, a maximum of eight epochs and patience of two; their learning rate is 1×10^{-3}. Early stopping selects the checkpoint with the best validation F1-score. The pretrained BERT models use linear learning-rate scheduling with warm-up over 6% of the total training steps. For each seed, the test partition is evaluated after model selection and training. The three seeds (42, 43 and 44) control Python, NumPy and PyTorch random states, data-loader ordering and the stratified data split; consequently, the test partitions can differ between runs. Within each run, every model is evaluated on the same test partition. Other sources of variation, including non-deterministic GPU operations, may remain.

5.6 Evaluation Protocol

Accuracy, precision, recall, F1-score and ROC-AUC are computed on the held-out test partition of each run, with Fake as the positive class: TP is a Fake article classified as Fake, FN a Fake article classified as Real, FP a Real article classified as Fake and TN a Real article classified as Real. ROC-AUC is computed from the predicted Fake-class probability. The false-negative rate is FNR = FN / (FN + TP). Table II reports, for each model, the mean and the sample standard deviation (ddof = 1) of each metric over the three runs; accuracy, precision, recall and F1-score are given in percent and ROC-AUC on its 0–1 scale. Table III reports confusion-matrix counts summed over the three test runs, which is different from the per-run averaging in Table II. No statistical significance test was performed; with three runs, the standard deviations indicate run-to-run variability only. The supplied code, the WELFake file and the settings above are intended to allow another researcher to repeat the procedure.

Fig. 2. Pseudocode of the implemented training and prediction pipeline (Algorithm 1).

```
Algorithm 1: Implemented WELFake Training and Prediction Pipeline
Input:  WELFake CSV D (title, text, label); pretrained BERT B; seed s
Output: fake-class probability p; label ŷ and confidence c may be derived from p
D ← LoadWELFake(D); missing title ← empty string
D ← { a ∈ D : length(a.text) ≥ 50 characters }
x_a ← a.title + a.text for each a; D ← DropExactDuplicates(D, x)
y_a ← a.label                  // 0 = Real, 1 = Fake
(D_tr, D_va, D_te) ← StratifiedSplit(D, 80/10/10, seed s)
Fit training-only TF-IDF vectoriser, 200-component SVD and StandardScaler; apply fitted transforms to D_va and D_te
for each article x in a batch do
    T ← BertTokenize(x, max_len = 256)    // uncased WordPiece
    H ← B(T)                              // token representations
    h_cls ← H[CLS]                        // document-level vector
    h_cnn ← MaxPool(Conv1D_{k = 2,3,4; 128 filters each}(H))  // CNN over BERT outputs
    f ← [SVD(TFIDF(x)); LinguisticFeatures(x)]  // 7 features
    h_aux ← AuxNet(f)  // 256-ReLU-Dropout(0.3)-128-ReLU
    h ← Dropout(0.1, LayerNorm([h_cls; h_cnn; h_aux]))
    z ← Linear(h);  p ← σ(z)              // p = P(Fake)
    ŷ ← Fake if p ≥ 0.5 else Real;  c ← max(p, 1 − p)
    return p  // derive ŷ by thresholding p; c = max(p, 1 − p)
end for
Training: minimise BCEWithLogitsLoss on D_tr with AdamW; early-stop on validation F1; evaluate D_te once per model/run
```

6. EXPERIMENTAL RESULTS AND DISCUSSION

This section reports and discusses the results of the experiments described in Section 5. Table II gives the mean and sample standard deviation of each metric over seeds 42, 43 and 44, Table III gives the confusion-matrix counts of the proposed hybrid summed over the three test runs, and Figs. 3–6 were regenerated from the per-run results file; they use the same values as Table II. All models shared a test partition within each seed; because the split is regenerated per seed, test articles may overlap across runs.

TABLE II. TEST-SET PERFORMANCE: MEAN ± SAMPLE SD OVER SEEDS 42, 43, 44 (ACCURACY, PRECISION, RECALL, F1 IN %; ROC-AUC ON 0–1 SCALE)

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Naive Bayes	90.01 ± 0.25	87.45 ± 0.45	90.61 ± 0.04	89.00 ± 0.25	0.96438 ± 0.00246
Logistic Regression	96.84 ± 0.47	96.48 ± 0.73	96.43 ± 0.35	96.46 ± 0.52	0.99533 ± 0.00106
SVM (linear)	97.02 ± 0.28	96.71 ± 0.43	96.61 ± 0.23	96.66 ± 0.31	0.99568 ± 0.00068
CNN (word embeddings)	95.94 ± 0.39	94.88 ± 0.81	96.11 ± 1.74	95.48 ± 0.49	0.99358 ± 0.00109
LSTM	96.01 ± 0.25	94.24 ± 0.56	96.98 ± 0.10	95.59 ± 0.26	0.99304 ± 0.00116
BERT (fine-tuned)	99.45 ± 0.14	99.42 ± 0.41	99.35 ± 0.27	99.38 ± 0.15	0.99984 ± 0.00004
Proposed Hybrid (BERT + CNN/Linguistic)	99.62 ± 0.06	99.64 ± 0.00	99.51 ± 0.13	99.58 ± 0.07	0.99968 ± 0.00007

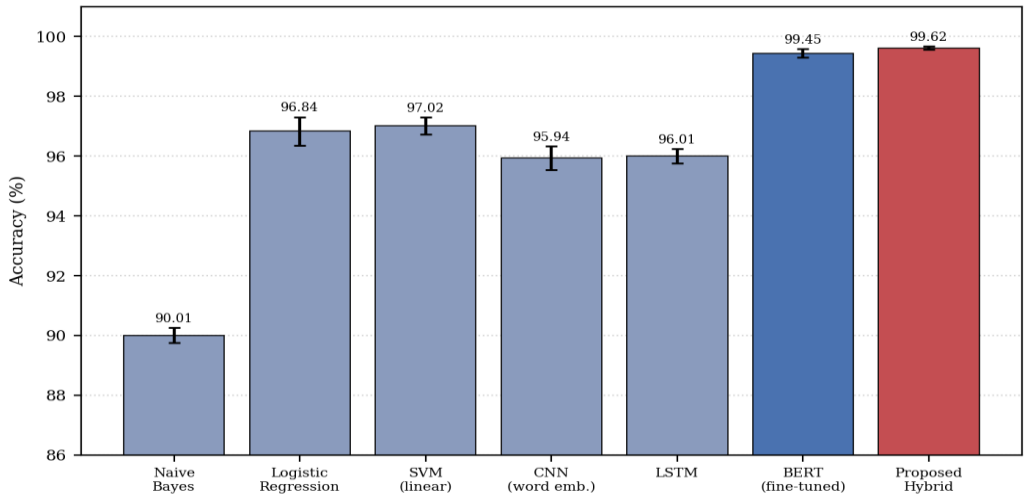


Fig. 3. Test-set accuracy by model (mean ± sample SD over seeds 42, 43 and 44; WELFake). The y-axis is truncated at 86%.

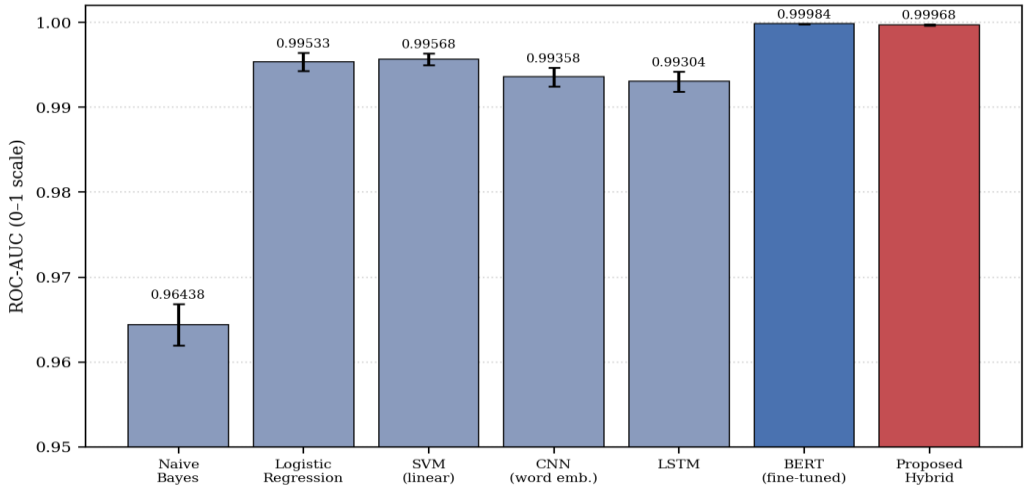


Fig. 4. Test-set ROC-AUC by model on the 0–1 scale (mean ± sample SD over three seeds; WELFake). The y-axis is truncated at 0.95.

TABLE III. CONFUSION-MATRIX COUNTS FOR THE PROPOSED HYBRID, SUMMED OVER THE THREE TEST RUNS (18,852 EVALUATION INSTANCES)

	Predicted: Fake	**Predicted: Real**
Actual: Fake	TP = 8374	FN = 41
Actual: Real	FP = 30	TN = 10407

TP = True Positive (fake correctly identified), FN = False Negative (fake misclassified as real), FP = False Positive (real misclassified as fake), TN = True Negative (real correctly identified). Fake is the positive class.

Each test partition contains 6,284 articles (2,805 Fake and 3,479 Real), so the table covers 18,852 evaluation instances, which are not necessarily distinct articles (Section 5.2). The pooled Fake recall is 8,374/8,415 = 99.51% and the pooled precision is 8,374/8,404 = 99.64%, consistent with the means in Table II; the pooled false-negative rate is 41/8,415 = 0.49% and the pooled false-positive rate is 30/10,437 = 0.29%.

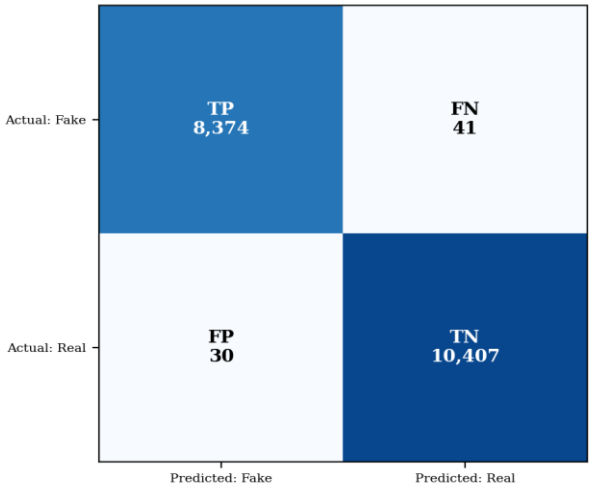


Fig. 5. Confusion matrix of the proposed hybrid, summed over three test runs (WELFake); corresponds to Table III.

The hybrid obtained higher mean accuracy, precision, recall and F1-score than fine-tuned BERT over the three runs (99.62% vs. 99.45% accuracy; 99.64% vs. 99.42% precision; 99.51% vs. 99.35% recall; 99.58% vs. 99.38% F1-score; Table II, Fig. 6). At the run level, the hybrid was higher in all three runs for accuracy, recall and F1-score, but its precision was lower than that of BERT in the run with seed 43. The hybrid's mean ROC-AUC (0.99968 ± 0.00007) was lower than that of BERT (0.99984 ± 0.00004) in all three runs, so the hybrid is not superior on every metric. All differences are small, the error bars in Fig. 6 overlap, only three runs were used and no significance test was performed; the results do not show that the hybrid is statistically better than BERT.

Among the baselines, the linear SVM obtained the highest mean accuracy (97.02%) and ROC-AUC (0.99568), with logistic regression close behind (96.84%), while the CNN (95.94%) and the LSTM (96.01%) were slightly lower and Naive Bayes was lowest (90.01%, with precision 87.45%). The LSTM had the highest recall among the non-BERT models (96.98%) but lower precision (94.24%) than the linear models. Fine-tuned BERT improved on all of these models by more than two percentage points of accuracy. In this experiment, linear classifiers therefore remained competitive with the CNN and LSTM baselines; this statement concerns these implementations and this dataset only. High scores on WELFake do not imply that any of these models would generalise to unseen publishers, topics or languages.

For fake-class detection (objective O2), the pooled false-negative rates over the three test runs were 9.39% for Naive Bayes, 3.56% for logistic regression, 3.39% for the linear SVM, 3.89% for the CNN, 3.02% for the LSTM, 0.65% (55 of 8,415 Fake articles) for fine-tuned BERT and 0.49% (41 of 8,415) for the hybrid. Corresponding false-positive counts were 49 for BERT and 30 for the hybrid out of 10,437 Real articles. These are aggregate counts over three runs whose test sets may overlap.

Variability across seeds is reported as the sample standard deviation of three values. With so few runs, these standard deviations are rough indications of stability and do not support confidence intervals or claims of significant improvement. The experiments reported here do not include an ablation study, an error analysis, inference-latency measurements or a cross-dataset evaluation; these omissions are described as limitations in Sections 7.8–7.11.

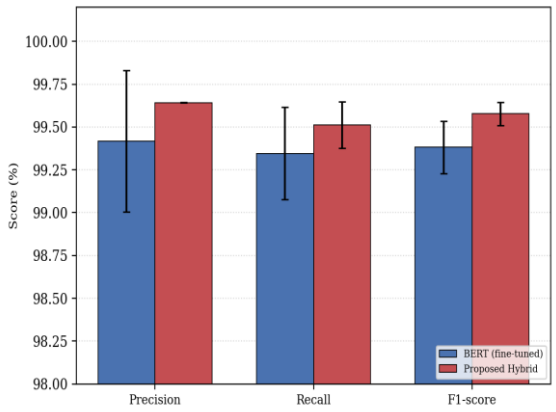


Fig. 6. Precision, recall and F1-score of fine-tuned BERT and the proposed hybrid (mean ± sample SD over three seeds; y-axis truncated at 98%). The hybrid's means are higher, but the error bars overlap.

7. ETHICAL CONSIDERATIONS AND LIMITATIONS

7.1 Dataset Bias

Fake-news datasets are often built from a small number of sources or periods, which can introduce publisher, topic and time-period bias. A classifier may then learn the style, vocabulary or topics of particular publishers, and not the factuality of the content, and a high score may reflect these shortcuts. WELFake merges four earlier collections [1], and the labels are attached to sources and not to verified claim-level judgements. Using several datasets can reduce this problem but does not remove it. Evaluation here was restricted to WELFake, so the effect of these biases on the model was not measured.

7.2 Explainability

Transformer-based models are often described as black boxes. The implemented pipeline has no explanation module (Section 4.5), so individual predictions cannot be explained, and this remains a limitation. Attribution methods such as SHAP or Integrated Gradients are possible future research directions; their outputs would indicate which inputs influenced the score and would not establish the causal reasons why an article is false.

7.3 Multilingual Limitations

The model was developed and evaluated on English-language text with an English-language pretrained encoder. Its behaviour on other languages is unknown. Extending it would require language-specific or multilingual transformer models and labelled datasets, which are not always available for other linguistic communities.

7.4 Adversarial Content

It is well documented in the NLP robustness literature that classifiers can be misled by crafted text. This can be done through text perturbations, paraphrasing or mixing true and false claims, and machine-generated text adds a further challenge [31]. No adversarial evaluation was performed, and malicious actors could exploit this weakness.

7.5 Misclassification Risk and Human Oversight

A false positive may label genuine reporting as fake and harm the reputation of legitimate authors; a false negative may allow misleading content to go undetected. The social costs of the two errors are not equal, so both should be reported and not only accuracy (Table III). Any moderation pipeline that relies on automated classification should include human verification for consequential decisions. The model is not a substitute for professional fact-checking, and it does not verify claims against external evidence.

7.6 Data Leakage and Evaluation Validity

Exact duplicates were removed before splitting, but near-duplicates were not, so partial overlap between partitions is possible. The selection of the hybrid's training settings was not documented (Section 5.4); selection by test-set results would inflate the reported numbers. The test sets of different seeds may overlap, so the three runs are not guaranteed to be independent samples.

7.7 Benchmark Performance and Real-World Reliability

High accuracy on a benchmark does not establish reliable fake-news detection in practice. The model may rely on publisher-specific or topic-specific patterns, news changes over time, and WELFake was evaluated in a single split-based setting. The results should therefore not be used to claim reliability on unseen publishers, topics, time periods or languages.

7.8 Ablation Study Not Performed

No ablation experiment was run. The implementation includes model variants that could isolate the convolutional branch and the auxiliary branch, but no results from them exist, and none are reported. The comparison between fine-tuned BERT and the hybrid shows the effect of all added components together, together with the differences in learning-rate and warm-up settings (Section 5.4); the separate contributions of the CNN branch, the TF-IDF/SVD features and the linguistic features, and the effect of the training settings, remain unverified. An ablation study isolating these components is future work (Section 8).

7.9 Error Analysis Not Performed

Only aggregate error counts are available (Table III): the hybrid produced 41 false negatives and 30 false positives over 18,852 evaluation instances. Saved predictions linked to article texts were not analysed, so no representative false-positive or false-negative examples are shown and no explanation of individual errors is offered. In general terms, genuine reporting that is sensational in style or resembles content from the fake-labelled sources may be flagged as fake, and carefully written fabricated content may be accepted as real; these are plausible limitations and not findings of this study. A qualitative analysis of the misclassified articles is future work (Section 8).

7.10 Inference Latency Not Measured

Inference latency was not measured, and the results do not show that the model meets any latency requirement for near-real-time use. Latency should be measured in future work under a specified device, batch size and sequence length (Section 8).

7.11 Cross-Dataset Evaluation Not Performed

Only WELFake was evaluated. Performance on other datasets, such as ISOT [9], FakeNewsNet [30] or LIAR [26], is unknown, and no claim of cross-dataset generalisation is made. Because WELFake merges several sources, evaluation on a collection that is independent of those sources would be an informative future test (see also Sections 7.1, 7.7 and 8).

8. CONCLUSION AND FUTURE WORK

This paper studied fake-news detection as binary text classification and evaluated a hybrid model that fuses a fine-tuned BERT encoder, a CNN over BERT token representations and an auxiliary branch of TF-IDF/SVD and seven linguistic features. The model was implemented and evaluated on WELFake (62,833 cleaned articles; 55.36% real and 44.64% fake) using a stratified 80/10/10 split repeated with seeds 42, 43 and 44, and was compared with Naive Bayes, logistic regression, a linear SVM, a word-embedding CNN, an LSTM and fine-tuned BERT. Over the three runs, the hybrid reached mean accuracy of 99.62%, precision of 99.64%, recall of 99.51% and F1-score of 99.58%, compared with 99.45%, 99.42%, 99.35% and 99.38% for fine-tuned BERT; its mean ROC-AUC (0.99968) was lower than that of BERT (0.99984). The differences between the two models are small, were not tested for statistical significance and are based on three runs. The main limitation is that the evaluation covers a single dataset, so the results do not show generalisation to other sources, topics, time periods or languages.

Future work includes: (i) ablation experiments isolating the CNN branch and the TF-IDF/SVD and linguistic-feature branch (Section 7.8); (ii) evaluation on at least one additional, structurally different dataset (Section 7.11); (iii) measurement of inference latency under a specified device and batch configuration (Section 7.10); (iv) prediction-level explainability, for example SHAP or Integrated Gradients, with validation of the explanations; (v) analysis of representative false positives and false negatives (Section 7.9); (vi) multilingual evaluation; (vii) improved robustness and generalisation, including near-duplicate removal and adversarial testing; and (viii) a possible link to external fact-verification resources [32]. Real-time moderation, multilingual detection, explanation generation and external fact verification were not implemented or tested in this study.

REFERENCES

[1] P. K. Verma, P. Agrawal, and R. Prodan, "WELFake dataset for fake news detection in text data," Zenodo, 2021, doi: 10.5281/zenodo.4561253.

[2] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," Science, vol. 359, no. 6380, pp. 1146–1151, 2018.

[3] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," J. Econ. Perspect., vol. 31, no. 2, pp. 211–236, 2017, doi: 10.1257/jep.31.2.211.

[4] N. Grinberg, K. Joseph, L. Friedland, B. Swire-Thompson, and D. Lazer, "Fake news on Twitter during the 2016 U.S. presidential election," Science, vol. 363, no. 6425, pp. 374–378, 2019.

[5] D. M. J. Lazer et al., "The science of fake news," Science, vol. 359, no. 6380, pp. 1094–1096, 2018, doi: 10.1126/science.aao2998.

[6] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," ACM SIGKDD Explor. Newsl., vol. 19, no. 1, pp. 22–36, 2017, doi: 10.1145/3137597.3137600.

[7] X. Zhou and R. Zafarani, "A survey of fake news: Fundamental theories, detection methods, and opportunities," ACM Comput. Surv., vol. 53, no. 5, Art. no. 109, pp. 1–40, 2020.

[8] R. Oshikawa, J. Qian, and W. Y. Wang, "A survey on natural language processing for fake news detection," in Proc. 12th Language Resources and Evaluation Conf. (LREC), 2020, pp. 6086–6093.

[9] 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, Springer, 2017, pp. 127–138.

[10] V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, "Automatic detection of fake news," in Proc. 27th Int. Conf. Computational Linguistics, 2018, pp. 3391–3401.

[11] B. 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 on News and Public Opinion, 2017, pp. 1–9.

[12] Y. Kim, "Convolutional neural networks for sentence classification," in Proc. EMNLP, 2014, pp. 1746–1751.

[13] N. Ruchansky, S. Seo, and Y. Liu, "CSI: A hybrid deep model for fake news detection," in Proc. ACM CIKM, 2017, pp. 797–806.
[14] J. A. Nasir, O. S. Khan, and I. Varlamis, "Fake news detection: A hybrid CNN-RNN based deep learning approach," Int. J. Inf. Manage. Data Insights, vol. 1, no. 1, 100007, 2021.
[15] 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, 2020, doi: 10.1016/j.cogsys.2019.12.005.
[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL-HLT, 2019, pp. 4171–4186.
[17] R. K. Kaliyar, A. Goswami, and P. Narang, "FakeBERT: Fake news detection in social media with a BERT-based deep learning approach," Multimed. Tools Appl., vol. 80, pp. 11765–11788, 2021, doi: 10.1007/s11042-020-10183-2.
[18] P. K. Verma, P. Agrawal, V. Madaan, and R. Prodan, "MCred: Multi-modal message credibility for fake news detection using BERT and CNN," J. Ambient Intell. Humaniz. Comput., published online Jul. 27, 2022, doi: 10.1007/s12652-022-04338-2.
[19] A. J. Keya, H. H. Shajeeb, M. S. Rahman, and M. F. Mridha, "FakeStack: Hierarchical Tri-BERT-CNN-LSTM stacked model for effective fake news detection," PLOS ONE, vol. 18, no. 12, e0294701, 2023, doi: 10.1371/journal.pone.0294701.
[20] F. Al-Quayed, D. Javed, N. Z. Jhanjhi, M. Humayun, and T. S. Alnusairi, "A hybrid transformer-based model for optimizing fake news detection," IEEE Access, vol. 12, pp. 160822–160834, 2024, doi: 10.1109/ACCESS.2024.3476432.
[21] P. K. Verma, P. Agrawal, I. Amorim, and R. Prodan, "WELFake: Word embedding over linguistic features for fake news detection," IEEE Trans. Comput. Soc. Syst., vol. 8, no. 4, pp. 881–893, 2021, doi: 10.1109/TCSS.2021.3068519.
[22] H. Ahmed, I. Traore, and S. Saad, "Detecting opinion spams and fake news using text classification," Security and Privacy, vol. 1, no. 1, e9, 2018.
[23] 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, 2019.
[24] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang, "Prominent features of rumor propagation in online social media," in Proc. IEEE 13th Int. Conf. Data Mining, 2013, pp. 1103–1108.
[25] X. Dong, U. Victor, and L. Qian, "Two-path deep semi-supervised learning for timely fake news detection," IEEE Trans. Comput. Soc. Syst., 2020 (preprint: arXiv:2002.00763).
[26] W. Y. Wang, "'Liar, liar pants on fire': A new benchmark dataset for fake news detection," in Proc. 55th Annu. Meeting Assoc. Comput. Linguistics, 2017, pp. 422–426.
[27] A. Zubiaga, M. Liakata, R. Procter, G. Wong Sak Hoi, and P. Tolmie, "Analysing how people orient to and spread rumours in social media by looking at conversational threads," PLoS ONE, vol. 11, no. 3, e0150989, 2016.
[28] F. K. A. Salem, R. Al Feel, S. Elbassuoni, M. Jaber, and M. Farah, "FA-KES: A fake news dataset around the Syrian war," in Proc. Int. AAAI Conf. Web and Social Media, vol. 13, 2019, pp. 573–582.
[29] W. Lifferth, "Fake News," Kaggle Dataset, 2018. [Online]. Available: https://www.kaggle.com/c/fake-news/data
[30] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "FakeNewsNet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media," Big Data, vol. 8, no. 3, pp. 171–188, 2020.
[31] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, "Defending against neural fake news," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 32, 2019.
[32] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, "FEVER: A large-scale dataset for fact extraction and VERification," in Proc. NAACL-HLT, 2018, pp. 809–819.