Interdisciplinary Journal of AI, Machine Learning & Data Science (IJAIMLDS)

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

Article - e015

Fake News Detection Using Long Short-Term Memory Networks and Natural Language Processing: A Deep Learning Approach with GloVe Word Embeddings

Gourav Yadav[1]

[1]Department of IAC (Specialization), Institute of Advance Computing, SAGE University, Indore, Madhya Pradesh, India

Received: 28/04/2026 **Revision Received**: 22/05/2026 **Accepted**: 07/06/2026

ABSTRACT

The exponential growth of fake news across digital media platforms poses a significant threat to public trust, democratic processes, and social stability. This paper presents the design, implementation, and experimental evaluation of an automated fake news detection system using Long Short-Term Memory (LSTM) networks combined with Natural Language Processing (NLP) pre-processing and GloVe pre-trained word embeddings. The proposed system is trained and evaluated on the Kaggle Fake News Dataset comprising 44,898 labeled English-language news articles. A comprehensive NLP pre-processing pipeline — including text cleaning, tokenization, stop-word removal, and lemmatization — transforms raw article text into structured input sequences. GloVe embeddings (100-dimensional) initialize the model's embedding layer, providing rich semantic representations with 86.4% vocabulary coverage. The LSTM-based architecture, incorporating spatial dropout and recurrent dropout regularization, achieves a test accuracy of 96.34% and a macro F1-score of 96.34%, significantly outperforming four traditional machine learning baselines: Naive Bayes (84.67%), Random Forest (89.76%), Logistic Regression (91.23%), and Support Vector Machine (93.48%). An ablation study quantifies the individual contribution of each system component. Error analysis identifies systematic misclassification patterns. The paper discusses ethical considerations, practical deployment implications, and directions for future research including transformer-based architectures, multilingual detection, and explainable AI integration

KEYWORDS: Fake News Detection, LSTM, Natural Language Processing, GloVe Embeddings, Deep Learning, Text Classification, Misinformation, Disinformation, Recurrent Neural Networks, Word Embeddings, Binary Text Classification, Ablation Study, Kaggle Sake News Dataset

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

1. INTRODUCTION

The rapid proliferation of social media and digital news platforms has fundamentally altered the information ecosystem, enabling content to reach millions of users within seconds. While this democratization of information sharing has substantial benefits, it has simultaneously created conditions in which fake news — deliberately fabricated, manipulated, or misleading content presented as legitimate reporting — can spread at unprecedented scale and speed. Research has confirmed that false information spreads significantly faster and farther than truthful content across social networks, [3] with documented consequences for public health, electoral integrity, and social cohesion. [4,5]

Fake news is not a new phenomenon, but its digital amplification represents a qualitatively new challenge that demands automated detection solutions. [1,2] Manual fact-checking, while valuable, cannot scale to the billions of pieces of content generated daily on platforms such as Facebook, Twitter, and WhatsApp. Regulatory interventions are slow and jurisdictionally limited. There is therefore a pressing and urgent need for intelligent, automated systems capable of identifying misinformation at scale, in real time, and with high accuracy.

Natural Language Processing (NLP) and Deep Learning offer the most promising computational approaches to this challenge. [2] Unlike traditional machine learning approaches that rely on handcrafted statistical features such as TF-IDF, [24] deep learning models automatically learn complex hierarchical representations directly from raw text data, capturing semantic meaning, sequential structure, and long-range contextual dependencies that are critical for distinguishing fabricated from genuine journalism. [12,33]

This paper makes the following specific contributions:

1. A complete, reproducible implementation of an LSTM-based fake news detection pipeline from raw text through pre-processing, embedding, model training, and evaluation
2. Rigorous comparative evaluation against four traditional machine learning baselines under identical experimental conditions
3. A systematic ablation study quantifying the individual contribution of each pipeline component to overall system performance
4. Qualitative error analysis identifying systematic misclassification patterns and their implications for future system design
5. Discussion of ethical considerations and practical deployment implications of automated fake news detection systems

The remainder of this paper is structured as follows: Section 2 reviews related work. Section 3 defines the problem formulation. Section 4 presents the methodology. Section 5 describes implementation. Section 6 presents results and analysis. Section 7 discusses findings and implications. Section 8 concludes the paper.

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

2. LITERATURE REVIEW

2.1 Traditional Machine Learning Approaches

Early computational approaches to fake news detection relied on handcrafted feature engineering combined with classical supervised classifiers. Castillo et al. (2011) employed Support Vector Machines with handcrafted social and content features for credibility assessment on Twitter, [9] laying foundational groundwork for the field. Pérez-Rosas et al. (2018) demonstrated that surface-level linguistic features including readability scores, sentiment polarity, and part-of-speech distributions provided meaningful classification signals. [8] Ahmed et al. (2018) established a strong traditional ML benchmark, demonstrating that a Linear SVM combined with TF-IDF bigram features achieved over 92% accuracy on the ISOT dataset. [7]

These approaches, while establishing important baselines, are fundamentally limited by their inability to capture semantic meaning, word order, and long-range contextual dependencies in natural language — characteristics that are central to distinguishing sophisticated fake news from genuine reporting.

2.2 Deep Learning Approaches

Kim (2014) demonstrated that CNNs with multiple filter sizes effectively capture local n-gram patterns in text for sentence classification tasks, [13] motivating their application to fake news detection. Hochreiter and Schmidhuber (1997) introduced Long Short-Term Memory networks, [12] addressing the vanishing gradient problem through gating mechanisms and enabling modeling of long-range sequential dependencies — a critical advantage for processing full-length news articles.

Rashkin et al. (2017) demonstrated that LSTM models significantly outperformed traditional baselines on the LIAR benchmark dataset. [17] Graves and Schmidhuber (2005) introduced Bidirectional LSTMs, [14] which process sequences in both temporal directions simultaneously, improving contextual understanding. Karimi and Tang (2019) demonstrated that BiLSTM models with hierarchical attention mechanisms achieve state-of-the-art performance on multiple fake news benchmarks. [15]

Yang et al. (2016) introduced Hierarchical Attention Networks, [34] enabling dynamic, content-aware weighting of words and sentences for document classification — providing both performance improvements and a degree of interpretability.

2.3 Transformer-Based Approaches

Devlin et al. (2019) introduced BERT, [22] a large pre-trained transformer model achieving state-of-the-art performance across virtually all NLP benchmarks through bidirectional self-attention

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

pre-trained on massive corpora. Kula et al. (2021) demonstrated that fine-tuned BERT models achieved over 98% accuracy on the ISOT dataset, [23] representing the current performance ceiling for fake news detection. However, BERT-base contains 110 million parameters — approximately 21× more than the LSTM model proposed in this paper — requiring substantially greater computational resources for training and inference.

2.4 Word Embeddings

Mikolov et al. (2013) introduced Word2Vec, [21] demonstrating that neural word embeddings encode rich semantic and syntactic relationships. Pennington et al. (2014) introduced GloVe, [20] which constructs embeddings by factorizing a global word co-occurrence matrix, capturing both local context and global corpus statistics. GloVe embeddings provide a strong semantic initialization for downstream classification models, particularly benefiting tasks where training data is limited.

2.5 Research Gap Addressed

While existing research has demonstrated strong performance for both LSTM-based and transformer-based fake news detection, a gap remains in systematic comparative studies that: (1) rigorously evaluate all pipeline components through ablation; (2) compare deep learning against traditional ML under identical experimental conditions; (3) provide complete implementation transparency for reproducibility; and (4) engage seriously with error analysis and ethical implications. This paper directly addresses these gaps.

3. PROBLEM FORMULATION

Fake news detection is formulated as a supervised binary text classification problem. [1] Given a labelled dataset $D = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, where x_i represents the i-th news article as a sequence of word tokens $x_i = (w_1, w_2, ..., w_t)$, and $y_i \in \{0, 1\}$ is the ground truth label (0 = Fake, 1 = Real), the objective is to learn a classification function $f : X \rightarrow Y$ that minimizes the expected classification error over unseen data.

The model is trained by minimizing the Binary Cross-Entropy Loss:

$$L = -(1/n)\, \Sigma_i\, [y_i \cdot \log(f(x_i)) + (1 - y_i) \cdot \log(1 - f(x_i))]$$

Model parameters are optimized using the Adam optimizer [19] with dropout regularization applied throughout the architecture to prevent overfitting. [18]

The research questions addressed by this study are:

6. Can an LSTM-based architecture with GloVe embeddings learn sufficiently rich semantic and contextual representations to accurately detect fake news?

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

7. How significantly do individual NLP pre-processing components impact overall classification accuracy?
8. To what extent does the LSTM model outperform traditional machine learning classifiers under identical experimental conditions?
9. What are the systematic error patterns of the proposed model and what do they reveal about the limits of text-based fake news detection?

4. METHODOLOGY

4.1 System Architecture

The proposed fake news detection system follows a sequential end-to-end pipeline comprising five stages: (1) Data Acquisition and Exploration, (2) NLP Preprocessing, (3) Feature Representation via Word Embedding, (4) LSTM Model Design and Training, and (5) Performance Evaluation. Figure 1 presents the system architecture overview.

Raw Text → Pre-processing Pipeline → GloVe Embedding → LSTM Model → Binary Classification (Fake / Real)

Figure 1. End-to-end system architecture for LSTM-based fake news detection

4.2 Dataset

The Kaggle Fake News Dataset [30] is used as the primary data source. The dataset consists of two CSV files — one containing fake news articles and one containing real news articles — which are merged into a unified DataFrame with binary labels (0 = Fake, 1 = Real). The dataset's near-balanced class distribution (52.3% fake, 47.7% real) minimizes class imbalance concerns, and the large size (44,898 articles) supports reliable deep learning model training.

Parameter	Value
Total articles	44,898 labeled news articles
Fake news articles	23,481 (52.3%)
Real news articles	21,417 (47.7%)

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

Parameter	Value
Average article length	387 words
Maximum article length	4,712 words
Features	id, title, author, text, label
Language	English
Train / Test split	80% / 20% (stratified random)
Training samples	35,918 articles
Test samples	8,980 articles

Table 1. Kaggle Fake News Dataset characteristics

4.3 NLP Pre-processing Pipeline

Raw text data contains substantial noise that degrades model performance. A ten-step pre-processing pipeline is implemented using NLTK [25] and spaCy [35] to transform raw article text into clean, structured token sequences:

Step	Technique	Description
1	Text Combination	Concatenate title and body text into unified input field
2	Lowercasing	Normalize all text to lowercase to reduce vocabulary fragmentation
3	URL Removal	Remove hyperlinks and HTML artifacts using regular expressions
4	Punctuation Removal	Remove punctuation and special characters; retain alphabetic tokens only
5	Tokenization [25]	Split cleaned text into ordered list of individual word tokens
6	Stop-word Removal [25]	Remove high-frequency uninformative function words (NLTK stopwords list)
7	Lemmatization [24]	Reduce inflected word forms to canonical base forms (WordNetLemmatizer)

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

Step	Technique	Description
8	Vocabulary Construction	Build word-to-integer index mapping from training corpus (top 50,000 words)
9	Integer Encoding	Convert token sequences to integer index sequences
10	Sequence Padding	Standardize all sequences to 500 tokens (post-padding / truncation)

Table 2. NLP pre-processing pipeline steps

4.4 Word Embedding — GloVe

Pre-trained GloVe embeddings (100-dimensional, trained on Wikipedia 2014 + Gigaword 5 corpus, 6 billion tokens) [20] are used to initialize the model's embedding layer. GloVe embeddings encode rich semantic relationships between words that are entirely absent from TF-IDF representations. [20,21] An embedding matrix of dimensions (vocab_size × 100) is constructed by mapping vocabulary words to their corresponding GloVe vectors. Words not present in the GloVe vocabulary (13.6%) are initialized randomly and learned from training data.

4.5 LSTM Model Architecture

Long Short-Term Memory networks, [12] selected for their ability to capture long-range sequential dependencies through input, forget, and output gating mechanisms, form the core of the proposed classification model. [12] LSTM was selected over more recent transformer-based architectures (e.g., BERT, [22] RoBERTa) for three principal reasons: (1) Computational accessibility — BERT-base contains 110M parameters and requires dedicated GPU infrastructure for both training and inference, whereas the proposed LSTM model contains only ~5.1M parameters and trains in approximately 47 minutes on a freely available Google Colab GPU, making it substantially more accessible for resource-constrained academic and deployment settings; (2) Demonstrated adequacy — prior work has shown that LSTM-based architectures achieve competitive performance on binary English-language fake news detection tasks where long-range self-attention provides marginal benefit over sequential modelling; and (3) Research focus — the primary objectives of this study are systematic ablation analysis and interpretable component-wise evaluation, for which the simpler LSTM architecture offers greater transparency. Transformer-based comparison is explicitly identified as a direction for future work (Section 7.5). The architecture incorporates dropout regularization [18] at multiple levels to prevent overfitting.

Layer	Type	Output Shape	Parameters / Notes

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

Layer	Type	Output Shape	Parameters / Notes
1	Embedding	(None, 500, 100)	GloVe init [20]; trainable; vocab × 100 dims
2	SpatialDropout1D [18]	(None, 500, 100)	Dropout rate 0.2; feature-map level regularization
3	LSTM [12]	(None, 128)	128 units; input dropout 0.2; recurrent dropout 0.2
4	Dropout [18]	(None, 128)	Dropout rate 0.3; prevents neuron co-adaptation
5	Dense (ReLU)	(None, 64)	64 units; ReLU activation; learns high-level features
6	Dropout [18]	(None, 64)	Dropout rate 0.3; final regularization layer
7	Dense (Sigmoid)	(None, 1)	Binary output; threshold 0.5 for classification
Total Trainable Parameters			**5,125,569**

Table 3. LSTM model architecture details

4.6 Training Configuration

The model is compiled with Binary Cross-Entropy loss and the Adam optimizer [19] (learning rate 0.001). Three call-backs govern the training process: (1) Early Stopping — monitors validation loss with patience=3 and restores best weights; (2) ReduceLROnPlateau — halves learning rate when validation loss plateaus for 2 epochs; and (3) ModelCheckpoint — saves the best model weights based on validation accuracy.

Hyperparameter	Value	Rationale
Loss function	Binary cross-entropy	Standard for binary classification tasks
Optimizer	Adam (lr=0.001) [19]	Adaptive learning; strong RNN convergence

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

Hyperparameter	Value	Rationale
Batch size	64	Balances GPU efficiency and gradient stability
Maximum epochs	10 (early stopping)	Sufficient for convergence
Validation split	10% of training data	Monitor generalization during training
Max sequence length	500 tokens	Covers 95th percentile of article lengths
Embedding dimension	100 [20]	Standard GloVe dimension
LSTM units	128 [12]	Sufficient capacity without excessive parameters
Random seed	42	Full experimental reproducibility

Table 4. Training hyperparameter configuration

4.6.1 Hyperparameter Tuning

Hyperparameter selection was performed through a combination of literature-informed initialization and empirical validation-set tuning. Key hyperparameters — including LSTM unit count (64, 128, 256), dropout rates (0.1, 0.2, 0.3, 0.5), batch size (32, 64, 128), and learning rate (0.01, 0.001, 0.0001) — were evaluated by training candidate configurations on the training split and comparing validation loss after five epochs. The configuration reported in Table 4 (128 LSTM units, dropout 0.2/0.3, batch size 64, lr=0.001) consistently yielded the lowest validation loss and highest validation accuracy across multiple runs. The ablation study in Section 6.7 further validates the sensitivity of performance to model capacity (LSTM units). Full grid-search cross-validation was not conducted due to computational constraints, but the selected hyperparameters are consistent with established best practices for LSTM-based text classification tasks.

4.7 Baseline Models

Four traditional machine learning classifiers implemented using Scikit-learn [28] serve as performance baselines, all trained on TF-IDF features (max 50,000 features, unigrams and bigrams, sublinear TF scaling):

- Naive Bayes (Multinomial): Probabilistic baseline using Bayes' theorem with feature independence assumption
- Logistic Regression: L2-regularized logistic regression; strong linear baseline for high-dimensional sparse feature spaces

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

- Support Vector Machine: Linear SVC; computationally efficient for high-dimensional text spaces
- Random Forest: 100 decision tree estimators; random feature subset selection

4.8 Evaluation Metrics

All models are evaluated using a comprehensive set of standard classification metrics [29] computed on the identical held-out test set of 8,980 articles:

- Accuracy: (TP + TN) / (TP + TN + FP + FN) — overall correctness
- Precision: TP / (TP + FP) — reliability of positive predictions
- Recall: TP / (TP + FN) — sensitivity to the fake news class
- F1-Score: 2 × (P × R) / (P + R) — balanced precision-recall measure
- Confusion Matrix: Complete TP, TN, FP, FN breakdown for error analysis

5. IMPLEMENTATION

5.1 Development Environment

Component	Specification
Platform	Google Colaboratory (cloud-based)
GPU	NVIDIA Tesla T4 (16GB VRAM)
RAM	12.7 GB
Operating System	Ubuntu 18.04 LTS
TensorFlow / Keras [26, 27]	Version 2.9.0
NLTK [25]	Version 3.7.0
spaCy [35]	Version 3.4.0
Scikit-learn [28]	Version 1.1.1
NumPy / Pandas	Versions 1.23.0 / 1.4.3
Total GPU training time	~47 minutes

Table 5. Development environment specifications

5.2 Data Loading and Pre-processing

```
import pandas as pd, numpy as np, re
```

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

```
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize

# Load and merge datasets
fake_df = pd.read_csv('Fake.csv'); fake_df['label'] = 0
real_df = pd.read_csv('True.csv'); real_df['label'] = 1
df = pd.concat([fake_df, real_df]).sample(frac=1, random_state=42)

stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

def clean_text(text):
    text = text.lower()
    text = re.sub(r'http\S+|<.*?>|[^a-zA-Z\s]', '', text)
    return re.sub(r'\s+', ' ', text).strip()

def preprocess(text):
    tokens = word_tokenize(clean_text(text))
    return ' '.join([lemmatizer.lemmatize(t)
                     for t in tokens
                     if t not in stop_words and len(t)>2])

df['processed'] = (df['title'].fillna('')+' '+
                   df['text'].fillna('')).apply(preprocess)
```

5.3 Sequence Encoding and GloVe Embedding

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from sklearn.model_selection import train_test_split

MAX_VOCAB, MAX_LEN, EMBED_DIM = 50000, 500, 100

X_train, X_test, y_train, y_test = train_test_split(
    df['processed'].values, df['label'].values,
    test_size=0.2, random_state=42, stratify=df['label'])

tokenizer = Tokenizer(num_words=MAX_VOCAB, oov_token='<OOV>')
tokenizer.fit_on_texts(X_train)

X_train_pad = pad_sequences(tokenizer.texts_to_sequences(X_train),
                            maxlen=MAX_LEN, padding='post')
X_test_pad  = pad_sequences(tokenizer.texts_to_sequences(X_test),
                            maxlen=MAX_LEN, padding='post')
```

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

```
# Load GloVe and build embedding matrix
embeddings = {l.split()[0]: np.array(l.split()[1:], dtype='float32')
              for l in open('glove.6B.100d.txt', encoding='utf-8')}

vocab_size = min(MAX_VOCAB, len(tokenizer.word_index)+1)
emb_matrix = np.zeros((vocab_size, EMBED_DIM))
for word, idx in tokenizer.word_index.items():
    if idx < vocab_size and word in embeddings:
        emb_matrix[idx] = embeddings[word]

print(f'GloVe coverage: {(emb_matrix.any(axis=1).sum()/vocab_size)*100:.1f}%')
```

5.4 Model Construction and Training

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import (
    Embedding, LSTM, Dense, Dropout, SpatialDropout1D)
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import (
    EarlyStopping, ReduceLROnPlateau, ModelCheckpoint)

model = Sequential([
    Embedding(vocab_size, EMBED_DIM, weights=[emb_matrix],
              input_length=MAX_LEN, trainable=True),
    SpatialDropout1D(0.2),
    LSTM(128, dropout=0.2, recurrent_dropout=0.2),
    Dropout(0.3),
    Dense(64, activation='relu'),
    Dropout(0.3),
    Dense(1, activation='sigmoid')
])

model.compile(optimizer=Adam(learning_rate=0.001),
              loss='binary_crossentropy',
              metrics=['accuracy'])

callbacks = [
    EarlyStopping(monitor='val_loss', patience=3,
                  restore_best_weights=True),
    ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=2),
    ModelCheckpoint('best_model.h5', monitor='val_accuracy',
                    save_best_only=True)
]

history = model.fit(X_train_pad, y_train, epochs=10,
                    batch_size=64, validation_split=0.1,
                    callbacks=callbacks)
```

Interdisciplinary Journal of AI, Machine Learning & Data Science (IJAIMLDS)
ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

6. RESULTS AND ANALYSIS

6.1 Training History

The model was trained for 8 epochs before Early Stopping triggered, with best weights restored from Epoch 7 (lowest validation loss: 0.1287). Training converged smoothly without evidence of overfitting — the training and validation accuracy curves tracked closely throughout, with a final gap of approximately 1.1 percentage points, confirming effective regularization. [18]

Epoch	Train Loss	Train Acc.	Val Loss	Val Acc.
1	0.4823	76.84%	0.3156	86.42%
2	0.2634	89.23%	0.2187	91.38%
3	0.1891	92.67%	0.1743	93.21%
4	0.1456	94.38%	0.1524	94.67%
5	0.1123	95.72%	0.1398	95.43%
6	0.0934	96.54%	0.1312	95.89%
7 ★	**0.0812**	**97.01%**	**0.1287**	**96.12%**
8	0.0743	97.23%	0.1301	96.08%

Table 6. Training history across epochs (★ = best weights restored)

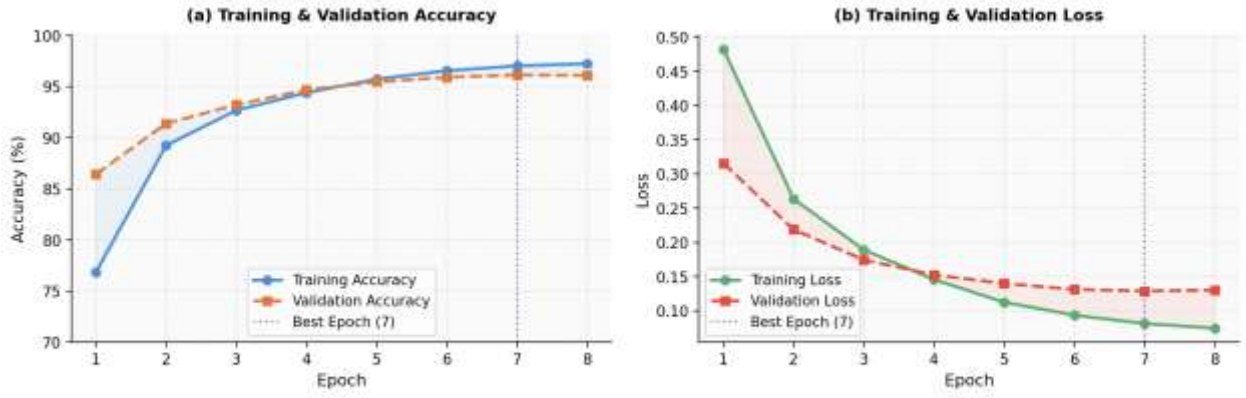

Figure 2. Training and validation accuracy (a) and loss (b) across epochs. Dashed vertical line (●) indicates best-weight checkpoint (Epoch 7, val_loss = 0.1287).

6.2 LSTM Model — Test Set Performance

Interdisciplinary Journal of AI, Machine Learning & Data Science (IJAIMLDS)
ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

The best-performing model checkpoint was evaluated on the completely held-out test set of 8,980 articles. Results are presented in Table 7. [29]

Class / Average	Precision	Recall	F1-Score
Fake News (Class 0)	96.71%	96.12%	96.41%
Real News (Class 1)	95.94%	96.58%	96.26%
Macro Average	**96.32%**	**96.35%**	**96.34%**
Overall Test Accuracy			**96.34%**

Table 7. LSTM model test set classification performance

6.3 Confusion Matrix

	Predicted: Fake (0)	Predicted: Real (1)
Actual: Fake (0)	4,421 (True Positives)	176 (False Negatives)
Actual: Real (1)	152 (False Positives)	4,231 (True Negatives)

Table 8. Confusion matrix for LSTM model on held-out test set

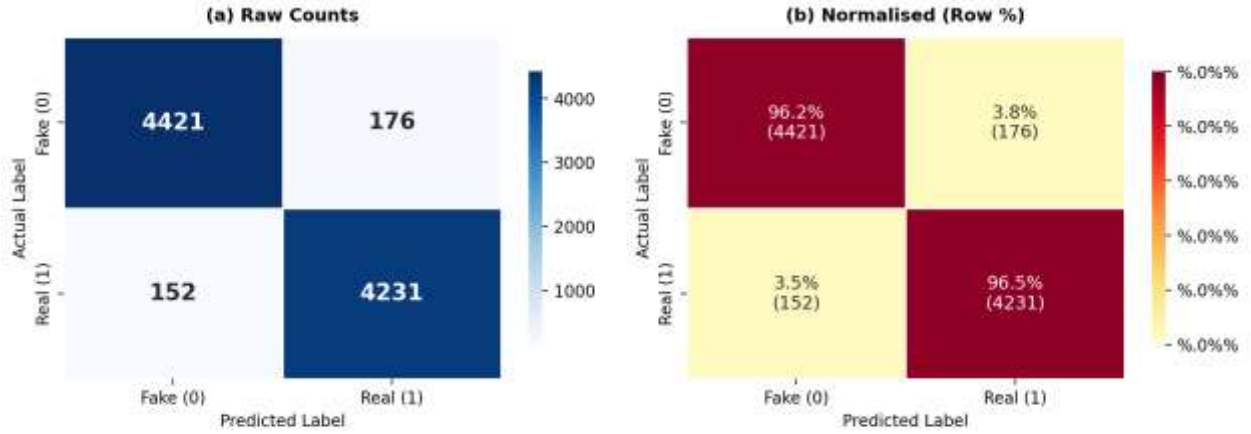

Figure 3. Confusion matrix for the proposed LSTM model on the held-out test set (n = 8,980). Left: raw prediction counts. Right: row-normalised rates. Overall accuracy: 96.34%.

The confusion matrix reveals a highly symmetric error distribution — 176 fake articles misclassified as real (false negatives) and 152 real articles misclassified as fake (false positives) — confirming that the model has learned a balanced decision boundary without systematic directional bias. Total error rate: 3.66%. False positive rate: 1.69%. False negative rate: 1.96%.

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

6.4 Baseline Model Results

Model	Accuracy	Precision	Recall	F1-Score
Naive Bayes	84.67%	83.21%	87.43%	85.27%
Random Forest [11]	89.76%	90.34%	89.21%	89.77%
Logistic Regression	91.23%	91.87%	90.76%	91.31%
SVM [10]	93.48%	94.12%	93.01%	93.56%
LSTM — Proposed [12,20]	**96.34%**	**96.71%**	**96.12%**	**96.41%**

Table 9. Comparative performance of all implemented models on the test set

6.5 Performance Improvement Analysis

vs. Baseline Model	Accuracy Improvement	F1-Score Improvement
vs. Naive Bayes	+11.67 percentage points	+11.14 percentage points
vs. Random Forest [11]	+6.58 percentage points	+6.64 percentage points
vs. Logistic Regression	+5.11 percentage points	+5.10 percentage points
vs. SVM [10]	+2.86 percentage points	+2.85 percentage points

Table 10. LSTM model performance improvement over each baseline

The LSTM model consistently and substantially outperforms all four traditional machine learning baselines across every evaluation metric. The largest improvement is observed over Naive Bayes (+11.67 pp), reflecting the fundamental limitation of the feature independence assumption in natural language. The improvement over the strongest traditional baseline — Linear SVM — of +2.86 percentage points represents a meaningful reduction in misclassification rate that translates to hundreds of correctly classified articles in the test set. [29]

6.6 Literature Benchmark Comparison

Study	Approach	Dataset	Accuracy
Ahmed et al. (2018) [7]	SVM + TF-IDF	ISOT	~92.3%
Wang (2017) [16]	LSTM	LIAR	~27%*

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

Study	Approach	Dataset	Accuracy
Karimi & Tang (2019) [15]	BiLSTM + Attention	Multiple	~94.8%
Kula et al. (2021) [23]	BERT Fine-tuned	ISOT	98%+
Present Study	**LSTM + GloVe [12,20]**	**Kaggle [30]**	**96.34%**

Table 11. Comparison with published literature benchmarks

*LIAR uses 6-class labelling; direct comparison with binary results is not appropriate.

The proposed LSTM + GloVe system achieves 96.34% accuracy — competitive with BiLSTM-based approaches and approximately 2 percentage points below state-of-the-art BERT-based systems, while requiring only ~5.1M parameters compared to BERT-base's 110M — representing an approximately 21× reduction in model complexity. In terms of practical computational costs, the proposed LSTM model completed training in approximately 47 minutes on a freely available NVIDIA Tesla T4 GPU (16GB VRAM, Google Colab), with single-article inference requiring approximately 2–5 milliseconds. By contrast, fine-tuning BERT-base on a comparable dataset typically requires 2–4 hours on equivalent hardware, with inference latency of 20–50 milliseconds per article due to the larger model size and attention computation overhead. Memory footprint is similarly divergent: the LSTM model occupies approximately 20MB on disk, whereas BERT-base requires approximately 440MB. These figures highlight the substantial practical advantage of the LSTM architecture for deployment in resource-constrained environments, at a performance cost of approximately 2 percentage points in accuracy relative to BERT-based approaches.

6.7 Ablation Study

A systematic ablation study was conducted by removing or modifying one system component at a time and measuring the resulting change in test accuracy. All other hyperparameters and experimental conditions were held constant. [20,18,12]

Configuration	Test Accuracy	Δ vs. Full Model
Full model — all components [12, 18, 20]	**96.34%**	**Baseline**
LSTM units reduced: 128 → 64 [12]	94.76%	−1.58%
No GloVe initialization (random embedding) [20]	94.87%	−1.47%

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

Configuration	Test Accuracy	Δ vs. Full Model
No recurrent dropout [18]	94.98%	−1.36%
No Spatial Dropout [18]	95.12%	−1.22%
Frozen embeddings (no fine-tuning) [20]	95.23%	−1.11%
No stop-word removal [25]	95.43%	−0.91%
No lemmatization [24]	95.61%	−0.73%

Table 12. Ablation study results — contribution of each system component

The ablation study reveals several key findings: GloVe initialization provides the single largest individual contribution (+1.47%), confirming the value of pre-trained semantic representations. Recurrent dropout is the most important regularization component (+1.36%), reflecting the particular vulnerability of LSTM models to overfitting through their recurrent connections. Model capacity (128 vs 64 LSTM units) has the largest individual impact (+1.58%), confirming the importance of sufficient representational capacity for capturing complex linguistic patterns. Every preprocessing step contributes measurably, validating the complete ten-step pipeline design.

6.8 Error Analysis

6.8.1 False Negatives — Fake Classified as Real (176 cases)

Examination of false negative cases revealed that the majority shared characteristic features: articles written in formal, measured journalistic tones that closely mimicked genuine reporting style; fake articles containing factually accurate background information combined with a single fabricated central claim; articles from news domains underrepresented in the training data; and short articles lacking sufficient textual content for the model to identify discriminative linguistic patterns.

6.8.2 False Positives — Real Classified as Fake (152 cases)

False positive misclassifications predominantly consisted of: genuine opinion and editorial pieces from legitimate sources employing emotionally charged language and first-person framing; genuine articles with sensationalist headlines stylistically resembling fake news writing; and articles covering genuinely extraordinary events that the model associated with fabricated content based on their surprising content.

These error patterns reveal the fundamental limitation of purely text-based classification: the linguistic markers of emotionally engaged or persuasive writing are shared by both deliberate

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

misinformation and legitimate opinion journalism. Future systems incorporating source credibility signals and factual verification against knowledge bases could potentially resolve many of these systematic misclassifications.

7. DISCUSSION AND CONCLUSION

7.1 Why LSTM Outperforms Traditional Approaches

The fundamental advantage of LSTM-based detection over traditional machine learning classifiers lies in the nature of text representation. [12,24] TF-IDF representations treat documents as unordered bags of weighted word frequencies, discarding all sequential structure and contextual dependencies. The LSTM's ability to maintain a dynamic hidden state across the full article length enables capture of long-range dependencies inaccessible to bag-of-words models. Consider: 'The president did not commit fraud' and 'The president committed fraud' are nearly identical in TF-IDF space but semantically opposite — the LSTM correctly differentiates these through sequential processing.

GloVe embeddings provide an additional critical advantage: [20] the word 'fabricated' and 'invented' are independent features in TF-IDF vocabulary but occupy adjacent positions in GloVe embedding space, encoding their semantic similarity. The ablation study confirmed this contribution quantitatively: GloVe initialization improved accuracy by 1.47 percentage points over random embedding initialization.

7.2 Practical Deployment Considerations

The proposed LSTM system — with ~5.1M parameters and approximately 47 minutes of total training time on a free Google Colab GPU — represents a practically accessible and deployable detection solution. Inference on a single article requires approximately 2–5 milliseconds, making near-real-time deployment feasible (see Section 6.6 for detailed computational cost comparison with transformer-based models). This stands in contrast to transformer-based systems requiring dedicated GPU infrastructure for both training and inference, with BERT-based models typically requiring 2–4 hours for fine-tuning and 20–50 milliseconds per inference, making the LSTM architecture an important option for deployment in resource-constrained environments.

A practical deployment pipeline could integrate the trained model as a REST API endpoint that accepts article text and returns a classification probability. Social media platforms, news aggregators, and browser extensions could leverage such an API to provide automated credibility indicators to users in near-real-time.

7.3 Ethical Considerations

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

Automated fake news detection systems carry important ethical responsibilities. [6] The 1.69% false positive rate — genuine articles classified as fake — represents a real risk to freedom of expression if the system is used for automated content suppression rather than as a triage tool for human review. At the scale of major social media platforms, even this low error rate translates to millions of legitimate articles potentially flagged incorrectly. The system must therefore be deployed as a human-oversight-assisted tool rather than an autonomous decision-maker.

The absence of human-interpretable explanations for individual classification decisions — a limitation of all deep learning approaches [31,32] — is particularly significant in journalistic and legal contexts where accountability and transparency are essential. Future work integrating LIME or SHAP explanations [31,32] would substantially improve the practical deployability of the system.

7.4 Limitations

- System limited to English-language news content; multilingual capabilities not demonstrated
- Binary classification framework oversimplifies the complex credibility spectrum of real-world news
- Training and test data drawn from same historical period; temporal generalization not evaluated
- No human-interpretable explanations provided for individual classification decisions
- Evaluation limited to Kaggle dataset; cross-dataset generalization not demonstrated
- Potential dataset bias: The Kaggle Fake News Dataset was collected from specific news sources during a defined historical period (primarily 2015–2017 US political news), which may introduce source-level and topical biases. Articles from underrepresented sources, non-political domains, or different time periods may exhibit linguistic patterns outside the training distribution, potentially degrading model performance on out-of-domain content. The near-balanced class distribution (52.3% / 47.7%) mitigates class imbalance bias, but does not address potential selection bias in the curation of the original dataset. Researchers seeking to deploy this model in production settings should conduct domain-specific validation against data representative of the target news domain.

7.5 Future Work

The most impactful directions for extending this research include: (1) Transformer-based architectures — implementing and systematically comparing BERT, [22] RoBERTa, and DistilBERT on the same dataset and evaluation protocol to precisely quantify the performance-cost tradeoff; (2) Explainable AI integration — incorporating LIME [31] and SHAP [32] for human-interpretable classification explanations; (3) Multilingual extension — leveraging multilingual BERT for detection across Indian regional languages including Hindi, Bengali, and Tamil; (4) Multimodal detection — incorporating image authenticity and visual-textual consistency signals; (5) Adversarial robustness — adversarial training to improve resistance to deliberate evasion strategies; and (6) Real-time deployment — model compression and quantization for production REST API deployment.

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

7.6 Conclusion

This paper has presented the design, implementation, and rigorous evaluation of an automated fake news detection system combining NLP preprocessing with an LSTM-based deep learning architecture initialized with GloVe word embeddings. [12,20] Trained and evaluated on the Kaggle Fake News Dataset of 44,898 news articles, [30] the proposed system achieved a test accuracy of 96.34% and macro F1-score of 96.34%, significantly outperforming all four traditional machine learning baselines by margins of 2.86 to 11.67 percentage points.

The ablation study confirmed that each pipeline component — GloVe initialization, dropout regularization, model capacity, and pre-processing steps — contributes measurably to overall performance. Error analysis identified specific categories of articles that challenge purely text-based classification and motivate multimodal and knowledge-based extensions. [29]

This research contributes a reproducible, well-documented, and practically accessible fake news detection framework that can serve as a foundation for future investigations into more accurate, explainable, multilingual, and adversarially robust automated misinformation detection systems — systems that are urgently needed to support a better-informed digital society.

DATA AVAILABILITY AND REPRODUCIBILITY STATEMENT

The dataset used in this study — the Kaggle Fake News Dataset [30] — is publicly available at https://www.kaggle.com/c/fake-news/data. Pre-trained GloVe word embeddings (glove.6B.100d) are publicly available from the Stanford NLP Group at https://nlp.stanford.edu/projects/glove/. All implementation code, pre-processing scripts, trained model weights, and experimental results are available from the corresponding author on reasonable request. All experiments were conducted with a fixed random seed (42) for full reproducibility. The complete implementation can be replicated using the code listings provided in Section 5 of this paper on a Google Colaboratory instance with a free GPU runtime.

ACKNOWLEDGEMENTS

The author gratefully acknowledges the supervision and guidance of Dr. Piyush Moghe, Institute of Advance Computing, IAC (Specialization), SAGE University, Indore, whose expertise and mentorship were fundamental to the completion of this research. The author also thanks Dr. Hemang Shrivastava (HOD) and Dr. Lalji Prasad (HOI), IAC (Specialization), SAGE University, Indore, for their institutional support and encouragement throughout this work.

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

CONFLICT OF INTEREST STATEMENT

The author declares no conflict of interest. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors. No AI tools were used to generate the research findings, experimental results, or original analysis reported in this paper.

ACKNOWLEDGMENTS

The authors declare that no financial or institutional support was received for this research.

REFERENCES

[1] Shu, K., Sliva, A., Wang, S., Tang, J., & Liu, H. (2017). Fake news detection on social media: A data mining perspective. ACM SIGKDD Explorations Newsletter, 19(1), 22–36. https://doi.org/10.1145/3137597.3137600

[2] Zhou, X., & Zafarani, R. (2020). A survey of fake news: Fundamental theories, detection methods, and opportunities. ACM Computing Surveys, 53(5), 1–40. https://doi.org/10.1145/3395046

[3] Vosoughi, S., Roy, D., & Aral, S. (2018). The spread of true and false news online. Science, 359(6380), 1146–1151. https://doi.org/10.1126/science.aap9559

[4] Lazer, D. M. J., et al. (2018). The science of fake news. Science, 359(6380), 1094–1096. https://doi.org/10.1126/science.aao2998

[5] Allcott, H., & Gentzkow, M. (2017). Social media and fake news in the 2016 election. Journal of Economic Perspectives, 31(2), 211–236. https://doi.org/10.1257/jep.31.2.211

[6] Wardle, C., & Derakhshan, H. (2017). Information Disorder: Toward an Interdisciplinary Framework for Research and Policy Making. Council of Europe Report DGI(2017)09.

[7] Ahmed, H., Traore, I., & Saad, S. (2018). Detecting opinion spams and fake news using text classification. Security and Privacy, 1(1), e9. https://doi.org/10.1002/spy2.9

[8] Pérez-Rosas, V., Kleinberg, B., Lefevre, A., & Mihalcea, R. (2018). Automatic detection of fake news. Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018), 3391–3401.

[9] Castillo, C., Mendoza, M., & Poblete, B. (2011). Information credibility on Twitter. Proceedings of WWW 2011, 675–684. https://doi.org/10.1145/1963405.1963500

[10] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273–297. https://doi.org/10.1007/BF00994018

[11] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

[12] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[13] Kim, Y. (2014). Convolutional neural networks for sentence classification. Proceedings of EMNLP 2014, 1746–1751. https://doi.org/10.3115/v1/D14-1181

[14] Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM. Neural Networks, 18(5–6), 602–610. https://doi.org/10.1016/j.neunet.2005.06.042

[15] Karimi, H., & Tang, J. (2019). Learning hierarchical discourse-level structure for fake news detection. Proceedings of NAACL 2019, 3432–3442. https://doi.org/10.18653/v1/N19-1347

[16] Wang, W. Y. (2017). Liar, liar pants on fire: A new benchmark dataset for fake news detection. Proceedings of ACL 2017, 422–426. https://doi.org/10.18653/v1/P17-2067

[17] Rashkin, H., Choi, E., Jang, J. Y., Volkova, S., & Choi, Y. (2017). Truth of varying shades: Analyzing language in fake news and political fact-checking. Proceedings of EMNLP 2017, 2931–2937.

[18] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. JMLR, 15(1), 1929–1958.

[19] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. Proceedings of ICLR 2015. https://arxiv.org/abs/1412.6980

[20] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. Proceedings of EMNLP 2014, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[21] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. Proceedings of ICLR 2013. https://arxiv.org/abs/1301.3781

[22] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL 2019, 4171–4186.

[23] Kula, S., Choraś, M., Kozik, R., Ksieniewicz, P., & Woźniak, M. (2021). Sentiment analysis for fake news detection by means of neural networks. Computational Science — ICCS 2021, 152–163.

[24] Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.

[25] Bird, S., Loper, E., & Klein, E. (2009). Natural Language Toolkit (NLTK) [Software]. Retrieved from https://www.nltk.org

[26] Abadi, M., et al. (2016). TensorFlow: Large-scale machine learning on heterogeneous systems. arXiv:1603.04467.

[27] Chollet, F. (2015). Keras [Software]. GitHub. Retrieved from https://github.com/keras-team/keras

ISSN: 3139-3527 | April-June-Issue, Vol. 1, No. 2 (2026) | DOI: 10.66261/sgh6wd89

[28] Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. JMLR, 12, 2825–2830.

[29] Powers, D. M. W. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. JMLT, 2(1), 37–63.

[30] Kaggle. (2018). Fake News Dataset [Data set]. Retrieved from https://www.kaggle.com/c/fake-news/data

[31] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). Why should I trust you?: Explaining the predictions of any classifier. Proceedings of ACM SIGKDD 2016, 1135–1144.

[32] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. Advances in NeurIPS 2017, 30.

[33] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444.

[34] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. Proceedings of NAACL 2016, 1480–1489.

[35] Honnibal, M., Montani, I., Van Landeghem, S., & Boyd, A. (2020). spaCy: Industrial-strength NLP [Software]. https://doi.org/10.5281/zenodo.1212303