Charmo Journal of Natural Sciences and Technologies (CJNST)

CJNST 2025, Vol. 0, Issue 1, pp. 1–11
https://cjnst.chu.edu.iq/

Research Article:

A Hybrid AI Model for Fake News Detection: Leveraging FastText and LSTM for Kurdish and English

Azad M. Karim[1*], Bryar A. Hassan[2,1]

[1]Department of Computer Science, College of Science, Charmo University, 46023 Chamchamal/Sulaimani, KRI, Iraq
[2]Computer Science and Engineering Department, University of Kurdistan Hewler, Erbil, Iraq

https://doi.org/10.31530/cjnst.2025.1.1

Received: 19 June 2025
Revised: 28 July 2025
Accepted: 20 August 2025

*** Corresponding author info:** Azad M. Karim; azad.mstafa@chu.edu.iq

Abstract.
Background: The spread of misinformation on digital platforms has created a pressing need for effective fake news detection (FND), particularly in low-resource languages such as Kurdish.
Aims: This study aimed to develop and evaluate a hybrid FastText–LSTM model for fake news detection in Kurdish and English, and to compare its performance with traditional machine learning (ML) and deep learning (DL) approaches.
Methodology: The Kurdish Fake News Dataset (KDFND) was preprocessed, balanced, and used to train multiple models, including Logistic Regression, SVM, Random Forest, and LSTM. The proposed hybrid model combined FastText embeddings with LSTM architecture and was tested on both Kurdish and English text.
Results: The hybrid FastText–LSTM achieved superior accuracy, with 94.25% for Kurdish and 92.11% for English, outperforming traditional ML models and standalone LSTM. Balanced data significantly improved classification results.
Conclusion: The hybrid FastText–LSTM model provides an effective solution for fake news detection in both high- and low-resource languages, establishing a strong baseline for Kurdish. Future work should explore transformer-based architectures and multilingual detection strategies.

Keywords: *Fake news detection, Machine learning, TF-IDF, Deep learning, Hybrid model, LSTM, FastText, Kurdish language processing.*

1. Introduction

The modern digital era allows news to spread because social media platforms exert widespread influence. While the ease of access to information has democratized sharing, it has also created a significant challenge: the accessibility of modern platforms has facilitated the spread of false and misleading information. Misinformation breeds division and disorder, erodes public trust, and diminishes accountability. The task of separating accurate news from misleading stories has become essential to our everyday lives because we face a massive influx of news reports that often contain false information.

Natural Language Processing (NLP) and Artificial Intelligence (AI) have made major progress on this critical task [1]. Researchers have developed numerous algorithms for fake news detection, thereby continuously increasing model accuracy. Undoubtedly, advancements in ML have made it easier to handle spam detection for high-resource languages (English, French, Arabic, etc.). However, it becomes considerably more complex in the case of low-resource languages (Kurdish) [2], where completely lack data. The scarcity of linguistic resources and limited research activity complicate fake news detection in the Kurdish language.

Methods such as LSTM and other DL alternatives performed even better, given their ability to consider sequential and contextual properties in text [3]. DL models, especially those combining word embeddings (like FastText) [4] with deep architectures, also offer excellent performance and have been applied successfully to low-resource languages such as Kurdish. The investigation of performance in different models for detecting fake news informs this study with the hybrid FastText and LSTM approach.

The aim of this study was to evaluate the effectiveness of various models in detecting fabricated news articles. This was achieved through a series of experiments conducted on a dataset of fake news articles written in Kurdish. The study contributes to enhancing the capacity for identifying fake news in the future. It employs specialized methods for detecting false information in both Kurdish and English, with a primary focus on Kurdish. Several computational models, including LR, MNB, RF, SVM, and others, were utilized to address this problem. The objective is to advance fake news detection in low-resource languages and to assess the effectiveness of different methodological approaches.

2. Related work

This section offers an in-depth summary of the advancements in FND methodologies, focusing on various ML and DL techniques.

Salh and Nabi (2023) [1] made major contributions to Kurdish FND by leveraging the KDFND dataset, which contains a balanced collection of 100,962 news articles. However, the specifics of the classifiers employed and their performance metrics were not disclosed, leaving a gap in understanding their effectiveness. On a more specialized level, Azad et al. (2021) [2] utilized an SVM for fake news classification in the Kurd Fake dataset.

They recorded commendable performance with 88.71% accuracy, which was complemented by the 88.71% precision, recall, and F1 scores. This study demonstrates the potential of traditional ML approaches in alleviating the challenges associated with low-resource languages such as Kurdish.

San Ahmed (2023) [5] took a more advanced approach by proposing a hard-voting ensemble technique that utilized SVM, NB, and DT classifiers. The novel approach yielded better results, as observed in 89.87% accuracy, 88.08% precision, 90.36% recall, and 89.11% F1-score. The findings demonstrate the effectiveness of ensemble techniques in boosting classification performance.

Sheikh et al. (2024) [6] made a contribution to the literature by conducting a comparative assessment with LR, achieving a very high accuracy of 96.87%. Its model also recorded the identical values of precision, recall, and F1-score, demonstrating its strong classification performance.

While most articles examined multiple ML and DL methods, some do not present explicit performance figures or data set details, and therefore comparison is difficult. Research by Bussa et al. (2023) [3], Ahmadi (2020) [7], and Taher et al. (2022) [8] addressed text processing techniques and classification algorithms but did not provide explicit evaluation metrics. Likewise, research by Hu et al. (2022) [9], Sadiq et al. (2023) [10], and Prachi et al. (2022) [11] examined DL techniques but did not have reported accuracy, precision, recall, or F1 scores.

More recent studies have also considered hybrid models, as found in the literature of Camelia and Fahim (2024) [12], Hashmi et al. (2024) [13], and Abdul et al. (2024) [14]. These publications describe the merging of conventional ML methods with DL structures to improve classification accuracy. However, similar to previous studies, there is no mention of performance measurements or dataset descriptions.

Recent advances in Transformer-based architectures, such as BERT, RoBERTa, and XLM-RoBERTa, have significantly improved the performance of natural language processing (NLP) tasks across a wide range of applications. These models leverage deep contextual representations and attention mechanisms to capture long-range dependencies in text more effectively than traditional methods. In particular, their multilingual and cross-lingual variants have demonstrated strong potential for fake news detection in diverse linguistic settings. Although this study primarily focuses on FastText and LSTM, incorporating Transformer-based models into future research may lead to further gains, especially for low-resource languages like Kurdish. For example, Saravanan et al. [4] employed a hybrid BERT-FastText model for depression detection, showcasing the synergy between pre-trained transformers and subword embeddings. Similarly, Hashmi et al. [13] explored hybrid and explainable AI approaches for multilingual fake news detection, achieving high accuracy. Moreover, Hu et al. [9] provide a comprehensive overview of DL methods for fake news detection, highlighting the transformative impact of Transformer architectures. Therefore, exploring such models represents a promising direction for expanding this work. Table 1 presents a comparative analysis of existing studies on fake news detection, outlining their focus areas, applied methodologies, and the corresponding research gaps.

Table 1: Comparative Analysis of Related Works in Fake News Detection

Ref	Focus Area	Methodology	Identified Gaps
[1]	Kurdish FND	Utilizes KDFND dataset	Lacks specifics on classifiers and performance metrics
[2]	Fake News Classification	SVM	Limited to SVM; lacks comparative analysis with other models
[5]	Ensemble Techniques	Hard Voting with SVM, NB, DT	No exploration of other ensemble methods or hybrid models

[6]	Comparative Assessment	LR	Does not compare with other ML models or datasets
[3], [7], [8]	Text Processing Techniques	Various algorithms	Lack of performance metrics hinders comparative analysis
[9], [10], [11]	DL Techniques	Various DL models	Absence of performance data limits understanding of effectiveness
[12], [13], [14]	Hybrid Models	Combination of ML and DL	No performance measurements or dataset descriptions provided

These gaps provide opportunities for future work to investigate alternative approaches, improve reporting transparency, and create hybrid models that successfully combine the best of conventional and contemporary ML techniques.

Several research gaps remain in this study. First, while the focus is on Kurdish and English, other low-resource languages remain underrepresented. Expanding the scope to include additional low-resource languages would enhance the generalizability of the findings. Second, although the KDFND dataset [15] is balanced, its relatively small size compared to datasets available for high-resource languages, such as English, may constrain the generalizability and robustness of the results.

3. The proposed model and experimental setup

3.1. Dataset

The KDFND [15] dataset consists of key attributes that facilitate the analysis and classification of Kurdish-language news articles into real or fake categories. The breakdown of the dataset is as shown in table 2.

Table 2: Summary of Attributes in the Kurdish Fake News Dataset (KDFND)

Attribute	Count
ID	90,905
Text	100,961
Text Translated to English	100,411
URL	100,800
Date	12
Source	103
Label	5
Unnamed Column	3

The dataset is filtered, cleaned, and labeled to ensure high-quality data for analysis. The primary focus is on the Text column, which contains the original Kurdish news articles, and the label column, among which 50,211 are fake and 50,751 are real news labeled as Fake and Real [1]. The classification of the article is either "Real" (0) or "Fake" (1).

3.2. Data Preprocessing Steps

Several preprocessing steps were applied to the dataset to ensure its cleanliness, balance, and suitability for ML algorithms. These steps are described below:

3.2.1. Removing Duplicates and Null Values:

Duplicate records were identified and removed to prevent redundancy. Dropped null values in the critical text and the English-translated text columns to maintain data integrity.

3.2.2. Balancing the Dataset:

The dataset initially exhibited class imbalance, with the majority of articles labeled as "Real." To address this, the majority class was “undersampled” to match the size of the minority "Fake" class, following the methodology of [14]. The balanced dataset was shuffled to minimize the influence of ordering bias during model training, as suggested in [5].

After preprocessing, the dataset was balanced to ensure equal representation of real and fake news articles. Below, Table 3 displays the balanced class distribution.

Table 3: Class Distribution in the Balanced Kurdish Fake News Dataset

Label	Count
0 (Real)	50210
1 (Fake)	50210

To create a dataset conducive to unbiased model training, performed balancing via “undersampling” of the majority class, following best practices from [2] and [3]. Consider Figure 1.

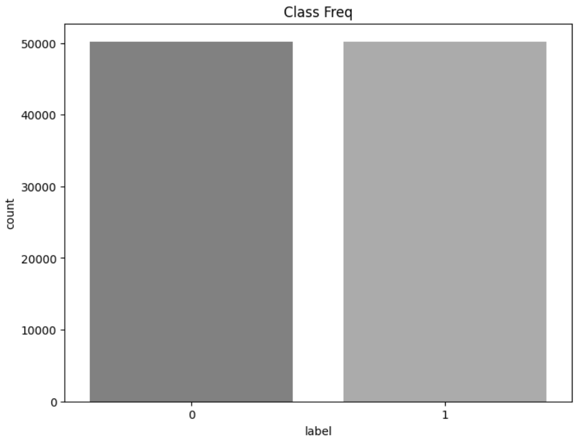

Figure 1: Balanced Labels of Dataset

This process is essential for improving model performance, and achieved better stability compared to the imbalance of a (3 -7) degree increase.

Table 4 presents the accuracy performance of various ML algorithms for Kurdish and English FND on imbalanced and balanced datasets. The models evaluated include LR, MNB, SGD, RF, SVM, and LSTM.

Table 4: Accuracy Comparison of Machine Learning Models on Balanced vs. Imbalanced Datasets (Kurdish and English)

Algo.	Imbalanced		Balanced	
	Kurdish Accuracy	**English Accuracy**	**Kurdish Accuracy**	**English Accuracy**
LR	81.20	79.15	83.31	79.15
MNB	79.66	76.63	82.49	76.63
SGD	79.67	78.06	81.41	78.06
RF	81.11	79.63	87.43	79.63
SVM	75.20	76.63	76.36	76.63
LSTM	83.50	79.91	87.70	84.32

3.2.3. Tokenization and Stopword Removal:

Tokenization was performed to split the text into individual words. Removed Kurdish stopwords and punctuation, including frequently occurring words that did not contribute to classification.

Tokenization splits text into discrete units (words, subwords, or characters). The process of tokenization, which is algorithmic rather than purely mathematical, yields the following output:

$$Tokenized\ Sequence: S = [w1, w2, \ldots, wn] \text{ ----------- (1)}$$

where *wi* is the *i*-th token in the sequence.

For subword tokenization (used in FastText), each word *wi* is further split into character *n-grams* (e.g., for "apple" with *n*=3: `'<ap', 'app', 'ppl', 'ple', 'le>'`).

The stopword list included commonly used Kurdish terms (e.g., "له" ",به" ",ئەو") and punctuation symbols, as guided by [7] and [11]. A frequency analysis of the Kurdish-language text revealed the most common words, many of which were identified as stopwords. These were removed to improve model performance. Additionally, rare words and outliers were examined to ensure they did not disproportionately affect the classification task.

3.2.4. Text Processing:

Irrelevant elements such as URLs, special characters, and numeric values were removed through regular expressions. Articles were stripped of extra spaces, and any resulting empty rows were eliminated.

3.2.5. Word Embedding and Representation:

Prepared the dataset for DL models by utilizing FastText embeddings to effectively capture word semantics. This functionality, like Glove embeddings have been shown to enhance NLP tasks, particularly in low-resource languages [9], [13]. The vocabulary size of the dataset was calculated. Assigned zero vectors to words not present in the embedding index to ensure compatibility with downstream deep-learning models. This embedding approach aligns with recent advancements in leveraging pre-trained word vectors for low-resource languages [8], [16].

GloVe Embeddings: GloVe learns embeddings by factorizing a word co-occurrence matrix. The objective function is

$$J = \sum_{i,j=1}^{v} f(Xij)\,(\boldsymbol{w}_i^T \boldsymbol{\acute{w}j} + bi + \acute{b}j - logXij)^2 \text{ --------- (2)}$$

where:

- *Xij* = co-occurrence count of words *i* and *j*.
- *f*(*Xij*) = weighting function (dampens rare/ frequent pairs).
- $\boldsymbol{wi}$, $\boldsymbol{\acute{w}j}$ = target and context embeddings.
- bi, $\acute{b}j$ = bias terms.

4. The proposed model

4.1. Machine Learning Models (MNB, LR, SVM, RF, and SGD)

The study implements several traditional ML models, including “MNB, LR, SVM, RF, and SGD,” for the task of Kurdish-language FND. Split the dataset into training (80%) and testing (20%) sets to evaluate the model on unseen data. The labels are converted to integers to facilitate model training and evaluation.

Construct a pipeline for each model, which incorporates text vectorization using "CountVectorizer," transformation using TF-IDF [17], [18] Transformer, and the corresponding ML algorithm.

TF-IDF (Term Frequency-Inverse Document Frequency): Used for text vectorization in ML models

$TF - IDF(t,d) = TF(t,d) \times IDF(t)$ ---------------- (3)

$IDF(t) = log\left(\frac{N}{DF(t)}\right)$ ----------------------------------- (4)

CountVectorizer: converts text into a document-term matrix where each entry represents the frequency of a term in a document

$Xij = count\ of\ term\ tj\ in\ document\ di$ --------------- (5)

where:

- TF(t,d) is the frequency of term t in document d.
- N being the total number of documents, and DF(t) the number of documents containing term t.
- Vocabulary: V={t1, t2,…,tm} be the set of unique terms in the corpus.
- Output: A sparse matrix, where n is the number of documents and mm is the size of V.

The pipeline ensures that the text data is preprocessed and transformed into a numeric format suitable for the model. The models are trained separately on the training data and predicted on the test data. The performance of the models is gauged using a classification report, which is accuracy. These metrics provide an overall assessment of the model in separating real and fake news articles correctly.

Use of conventional ML models is adequately evidenced in the current literature, e.g., the research of [1] and [2], which showed the performance of such models in Kurdish-language FND. Having more than one model facilitates comparison, as noted by [5], who emphasized the usefulness of ensemble techniques in enhancing detection accuracy.

4.2. LSTM Model

An LSTM network is built with the help of the Keras library to learn sequential patterns in text data. Text sequences are padded or truncated to a standard length of 100 tokens so that input shapes are standardized. Data is divided into training and validation sets, and labels are transformed into NumPy arrays of type float32 so that they are model-compatible.

The LSTM-only model uses GloVe embeddings, while the hybrid model uses FastText embeddings, aligned with the paper's main contribution. The model was trained using an 80/20 train-test split, a batch size of 64, and trained for 5 epochs with early stopping. A dense layer with a sigmoid activation function is used for binary classification. The model is compiled using binary cross-entropy loss and the Adam optimizer, with accuracy as the evaluation metric.

Binary Cross-Entropy Loss: Used for training the LSTM and hybrid models

$L(y,\hat{y}) = -\frac{1}{N}\sum_{i=1}^{N} y_i\, log(\hat{y}_i) + (1-y_i)log(1-\hat{y}_i)]$ -(6)

Adam Optimizer Update Rule: Used for model training

$\theta t + 1 = \theta t - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}}$ ----------------------------------(7)

Where

- $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected estimates of the first and second moments of gradients, and η is the learning rate.
- y_i is the true label and $\hat{y}_i$ is the predicted probability.

Early stopping and learning rate reduction callbacks are implemented to optimize training. Early stopping monitors validation loss and restores the best model weights, while the learning rate scheduler reduces the learning rate when validation loss plateaus. The model was trained for 10 epochs with a batch size of 64 and evaluate its performance on the validation set. Report the validation accuracy as a measure of the model's performance.

The use of LSTM for FND is supported by research such as that of [12], who demonstrated the effectiveness of LSTM in capturing contextual information in text. The integration of pre-trained embeddings aligns with the findings of [9], who emphasized the importance of leveraging pre-trained representations for improved performance in low-resource languages like Kurdish.

For the LSTM layer:

$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ --------------------- (8)

$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ ---------------------- (9)

$\hat{C}_t = tanh\,(W_c \cdot [h_{t-1}, x_t] + b_c)$ ---------------- (10)

$C_t = f_t \odot C_{t-1} + i_t \odot \hat{C}_t$ ------------------------- (11)

$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ ---------------------- (12)

$h_t = o_t \odot tanh\,(C_t)$ ------------------------------- (13)

where ft, it, ot are forget, input, and output gates; Ct is the cell state; and ht is the hidden state.

4.3. FastText Model

Employ the FastText model for FND, leveraging its ability to handle subword information and n-grams. Preprocess the data by adding the label to the text in the FastText training compatible format [19]. The data is divided into training and test datasets, and the training data is stored in a text file in FastText input format.

Train the "FastText" model using the "train_supervised" function with a learning rate of 0.5, 25 epochs, and word n-grams of dimension 2. The hyperparameters are selected to balance the complexity of the model and the training time. Test the model's performance on the test dataset and calculate evaluation measures such as precision, recall, and F1-score. The F1-score, which provides a balance between precision and recall, is of special concern for imbalanced datasets like FND.

The research conducted by sources [8] and [10] validates the use of "FastText," as these studies demonstrated its effectiveness in text classification, including various applications.ng FND. The use of n-grams allows the model to capture local word order and context, which is crucial in distinguishing between fake and real news.

FastText Embeddings: FastText represents each word as the sum of its subword (character n-gram) embeddings. For a word w with subwords $\{g1, g2, \ldots, gk\}$

$$Embedding(w) = \sum_{j-1}^{k} \boldsymbol{v}gj \text{ -------------------------- (14)}$$

where $\mathbf{v}gj$ is the embedding vector for subword *gj*.

- Training: Optimizes the skip-gram or CBOW objective (negative sampling or hierarchical softmax).

4.4. Hybrid FastText & LSTM Model

4.4.1. Data Preprocessing:

The data is preprocessed to make it compatible with FastText. Add the labels to the text, using "__label__0" for the real news case and "__label__1" for the fake news case. The data is divided into a training dataset and a testing dataset, and the training data is stored in a text file in FastText input format. This step is performed for enabling the FastText model to learn useful representations from the text data.

4.4.2. FastText Vectorization:

Train a FastText model using the pre-processed training data. FastText's subword information and n-gram support make it especially suitable for morphologically rich languages such as Kurdish. Sentence-level embeddings are created for every article using the trained FastText model. The embeddings encapsulate the semantic meaning of the text and are used as input features to the LSTM model.

4.4.3. Text Tokenization and Padding:

The text is tokenized by a tokenizer, which splits words into sequences of integers. Sequences are padded to a constant length to provide the LSTM model with fixed input shapes. Padding is required since LSTM needs fixed-length input sequences.

4.4.4. Embedding Matrix Construction:

FastText word vectors are then used to build an embedding matrix for the tokenized vocabulary. The matrix is a mapping of each word in the vocabulary to its FastText vector. The embedding layer initialized with FastText vectors is frozen during LSTM training to retain pre-trained subword features.

Embedding Matrix Construction: For a vocabulary V and embedding dimension d, the embedding matrix $\boldsymbol{E} \in \boldsymbol{R}^{|\boldsymbol{V}|\times \boldsymbol{d}}$ is:

$$E = \begin{bmatrix} vw1 \\ vw2 \\ \vdots \\ v_{w_{|v|}} \end{bmatrix} \text{ -------------------------------------- (15)}$$

where $\mathbf{v}wi$ is the pre-trained embedding (from FastText/GloVe) for word *wi*. Words not in the pre-trained vocabulary are typically mapped to a zero vector or a random vector.

Hybrid FastText-LSTM Embedding Input: The input to the LSTM is a sequence of FastText embeddings for each token

$$Input\ Sequence: [Embedding(w_1), Embedding(w_2), \ldots, Embedding(w_n)] \text{ ------- (16)}$$

where Embedding(*wi*) is the FastText vector for token *wi*.

4.4.5. Hybrid Model Architecture:

The hybrid model comprises an embedding layer initialized with the "FastText" embedding matrix.

followed by LSTM layers and dropout for regularization. The LSTM layers learn sequential dependencies in the text data, and the dropout layers avoid overfitting. Binary classification is done through dense layers with ReLU and sigmoid activations. The model was trained using an 80/20 train-test split, a batch size of 64, and 5 epochs, employing early stopping and the Adam optimizer with binary cross-entropy loss, using accuracy as the measure. Figure 2 shows the structure of hybrid FastText and LSTM.

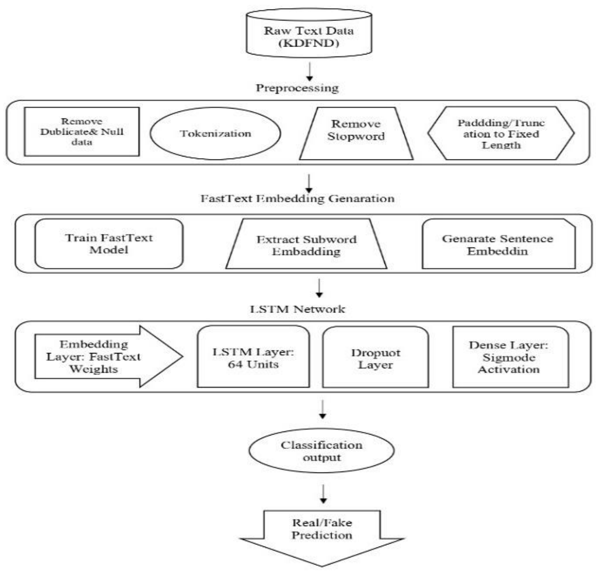

Figure 2: Hybrid FastText-LSTM for Fake News Detection

4.4.6. Model Training and Evaluation:

The model is trained for 3 epochs on the padded sequences in batches of 64. Early stopping can be added and learning rate reduction callbacks for improved training.

Evaluate the model based on its accuracy, F1-score, precision, and recall. These scores provide a general notion about how well the model detects real and fake news articles.

4.5. Pseudocode of the Hybrid FastText-LSTM model

Steps Explained:

1. Preprocessing (Aligned with Section 3.2):

- Noise removal, tokenization, and stopword filtering for Kurdish and English.
- Dataset balancing (undersampling) to address class imbalance.

2. FastText Embeddings (Aligned with Section 4.3):

- Trains FastText on the corpus to capture subword semantics (critical for Kurdish).
- Generates sentence embeddings by averaging word/subword vectors.

3. LSTM Architecture (Aligned with Section 4.4):

- Input: 100-dim FastText embeddings.
- LSTM layer (128 units) + Dropout for regularization.
- Sigmoid output for binary classification (Real=0/Fake=1).

4. Evaluation:

- Reports accuracy (primary metric in the paper).
- Supports precision/recall/F1 (Table 6 and Table 7) if needed. Pseudocode: Hybrid FastText-LSTM Model

```
# === 1. Data Preprocessing ===
def preprocess_text(text):
        text = remove_noise(text)                # Remove URLs, special chars, numbers
        tokens = tokenize(text)                  # Tokenize into words/subwords (for Kurdish/English)
        tokens = [word for word in tokens if word not in stopwords]   # Remove Kurdish/English stopwords
        return tokens
data = load_dataset("KDFND").                    # Load and balance dataset (undersampling)
data = balance_data(data)                        # 50,210 Real vs. 50,210 Fake (Table 3)
train_data, test_data = split_data(data, test_size=0.2)     # Split into train/test (80%/20%)
def train_fasttext(corpus):
# === 2. FastText Embeddings ===
    model = FastText.train(.                     # Train FastText with subword info (n-grams=2)
        sentences=corpus,
        vector_size=100,
        min_count=1,
        word_ngrams=2
    )
    return model
fasttext_model = train_fasttext(train_data["text"])
def get_embedding(text, model):                  # Generate sentence embeddings
        tokens = preprocess_text(text)
        embedding = mean([model[token] for token in tokens]).  # Average subword embeddings for each token
        return embedding
X_train = [get_embedding(text, fasttext_model) for text in train_data["text"]]
X_test = [get_embedding(text, fasttext_model) for text in test_data["text"]]
# === 3. LSTM Model ===
def build_lstm_model(input_dim):
        model = Sequential()
        model.add(InputLayer(input_shape=(input_dim,)))    # Input: FastText embeddings (100-dim)
        model.add(LSTM(128, return_sequences=False)).     # LSTM with 128 units + Dropout
        model.add(Dropout(0.5))
        model.add(Dense(1, activation="sigmoid")).     # Binary classification (Sigmoid)
        model.compile(                           # Compile with Adam optimizer
            loss="binary_crossentropy",
            optimizer="adam",
            metrics=["accuracy"]
        )
        return model
lstm_model = build_lstm_model(input_dim=100)
lstm_model.fit(
        X_train, train_data["label"],
        epochs=3,
        batch_size=64,
        validation_data=(X_test, test_data["label"])
)
# === 4. Evaluation ===
y_pred = lstm_model.predict(X_test) > 0.5.       # Threshold at 0.5
accuracy = (y_pred == test_data["label"]).mean()
print(f"Accuracy: {accuracy:.2%}")               # e.g., 94.25% (Table 7)
```

4.6. References Results

Table 5 is a comparison between different FND models, which have been tested on several different datasets and languages. It shows the accuracy, precision, recall, and F1 score of each method, along with some extra remarks concerning the method used.

4.6.1. Kurdish FND:

- CNN (Ref. [1]) achieves 91.60% accuracy on the KDFND, using a DL-based approach.
- SVM (Ref. [2]) performs at 88.71% accuracy on the KurdFake dataset, showcasing a traditional machine-learning approach for low-resource languages like Kurdish.
- Hard Voting (SVM + DT + NB) (Ref. [5]) improves performance with an accuracy of 89.87%, leveraging an ensemble approach.

4.6.2. Arabic FND:

FastText + DL (Ref. [8]) achieves 94.1% accuracy on an Arabic Fake News Dataset (Kaggle), demonstrating the effectiveness of word embeddings combined with DL.

4.6.3. English and General FND:

- FastText + DL (CNN) (Ref. [10]) applied to the Twitter Deepfake Dataset (TDD) attains 93.3% accuracy, showcasing its strength in detecting machine-generated tweets.
- LSTM (Ref. [12]) on the Fake News Challenge (FNC) Dataset achieves 97% accuracy, showing the potential of sequence-based DL for textual analysis.
- Logistic Regression (Ref. [6]) on the UCI Fake News Dataset reaches 96.87% accuracy, providing a traditional ML comparison against DL methods.
- XGBoost (Ref. [14]) applied to the ISOT Fake News Dataset achieves 99.67% accuracy, showcasing the power of boosting-based ML models for FND.

4.6.4. Multilingual FND:

Hybrid FastText + Explainable AI (Ref. [13]) leverages data from multiple prominent sources (Kaggle, McIntire, Reuters, and BuzzFeed Political), achieving a remarkable 99% accuracy across multiple languages. This emphasizes the possible advantages of hybrid AI models with explainability in FND.

Table 5: Cross-Language Model Comparison Based on Existing Studies

Ref.	Model/Method	Dataset	Language	Accuracy	Precision	Recall	F1-Score	Remarks
[1]	CNN	KDFND	Kurdish	91.60%	93.68%	97.34%	95.47%	Evaluated on Kurdish dataset using traditional ML models.
[2]	SVM	KurdFake (Kaggle)	Kurdish	88.71%	88.72%	88.71%	88.71%	Focused on low-resource Kurdish language.
[5]	Hard Voting (SVM + DT + 1NB)	KurdFake (Kaggle)	Kurdish	89.87%	88.76%	90.36%	89.55%	Ensemble approach for Kurdish FND.
[6]	LR	UCI Fake News Dataset	English	96.87%	96.95%	96.83%	96.85%	Comparative study of Naive Bayes and Logistic Regression for fake news.
[8]	FastText + DL	Arabic Fake News Dataset (Kaggle)	Arabic	94.1%	93.8%	94.3%	94.0%	Combined FastText embeddings with DL for Arabic fake news.
[10]	FastText + DL (CNN)	Twitter Deepfake Dataset (TDD)	English	93.3%	94.9%	95.5%	95.2%	Used FastText embeddings for detecting machine-generated tweets.
[12]	LSTM	Fake News Challenge (FNC) Dataset	English	97%	97%	98%	98%	Regularized LSTM for FND.
[13]	Hybrid FastText + Explainable AI	combined data from four prominent news datasets, including those from Kaggle, McIntire, Reuters, and BuzzFeed Political	Multilingual	99%	99%	99%	99%	Advanced hybrid model with explainable AI for improved performance.

[14]	ML (XGBoost)	ISOT Fake News Dataset	English	99.67%	99.45%	99.84%	99.64%	Comparative study of ML techniques for FND.

5. Discussion of Findings

Table 6 reports the results on the imbalanced dataset, while Table 7 presents the performance metrics on the balanced dataset of a hybrid FastText and LSTM model for FND using the Kurdish Fake News Dataset (KDFND). The model is evaluated separately for Kurdish and English texts, and its effectiveness is measured using accuracy, F1-score, precision, and recall. For Kurdish text, the model achieves (94.25%) accuracy for balanced and (91.42%) for imbalanced, indicating a high overall correctness in classification. For English text, the model performs slightly lower but is still effectively balanced (92.11%) and imbalanced (89.03%) with accuracy.

The results indicate that the FastText and LSTM hybrid model performs slightly better for Kurdish text compared to English, demonstrating its effectiveness in handling FND in a low-resource language like Kurdish.

Accuracy and F1-Score Reported in performance metrics (Table 6 and Table 7):

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} = \frac{TP+TN}{TP+TN+FP+FN} \text{ --------- (17)}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \text{ ---------------------------------- (18)}$$

$$Precision = \frac{TP}{TP+FP} \text{ -- (19)}$$

$$Recall = \frac{TP}{TP+FN} \text{ -- (20)}$$

Where:

- TP (True Positives): Correctly predicted "Fake" news.
- TN (True Negatives): Correctly predicted "Real" news.
- FP (False Positives): "Real" news incorrectly predicted as "Fake".
- FN (False Negatives): "Fake" news incorrectly predicted as "Real".

Table 6: Performance Metrics of Hybrid FastText-LSTM Model on Imbalanced KDFND Dataset

Model/Method	**Dataset**	Language	**Accuracy**	**Precision**	**Recall**	**F1-Score**
FastText and LSTM	KDFND	Kurdish	91.42%	88.68%	94.85%	91.66%
		English	89.03%	85.32%	94.14%	89.51%

Table 7: Performance Metrics of Hybrid FastText-LSTM Model on Balanced KDFND Dataset

Model/Method	**Dataset**	Language	**Accuracy**	**Precision**	**Recall**	**F1-Score**
FastText and LSTM	KDFND	Kurdish	94.25%	95.47%	92.91%	94.17%
		English	92.11%	92.35%	91.83%	92.09%

The Hybrid FastText + LSTM model outperforms previous Kurdish-specific models (e.g., CNN by Salh & Nabi (2023) [1] and SVM by Azad et al. (2021) [12]) with an accuracy of 94.25%, compared to 91.60% and 88.71%, respectively. A direct comparison with the work of Salh and Nabi [1], who also utilized the Kurdish Fake News Dataset (KDFND), highlights the advancements made in the current study. While their approach involved traditional ML techniques without reporting specific classifier types or detailed evaluation metrics, they achieved a respectable accuracy of 91.60% using DL. In contrast, our hybrid FastText-LSTM model applied to the same KDFND dataset achieved a higher accuracy of 94.25% on the balanced version and 91.42% on the imbalanced version. Additionally, our model reports comprehensive performance metrics, including precision, recall, and F1-score, offering a more thorough evaluation. The inclusion of FastText embeddings, which capture subword information, combined with LSTM's ability to model sequential dependencies, contributes to this improved performance. This comparison demonstrates the effectiveness of hybrid DL approaches over conventional or standalone methods for fake news detection in low-resource languages such as Kurdish.

The hybrid approach (FastText + LSTM) shows superior performance compared to traditional ML models (e.g., SVM, LR) and even some DL models (e.g., CNN) in both Kurdish and English datasets. While the study by Hashmi et al. (2024) [13] achieves a remarkable 99% accuracy across multiple languages using a hybrid model with understandable AI, it focuses more on low-resource languages like Kurdish, achieving 94.25% accuracy, which is still competitive.

The article does not explore transformer-based architectures (e.g., BERT, XLM-RoBERTa), which have shown higher accuracy in other studies (e.g., 97% for LSTM in English by Camelia & Fahim (2024) [12]).

The KDFND dataset used is smaller compared to datasets used in other studies (e.g., ISOT Fake News Dataset), which may affect the generalizability of the results.

5.1. Findings

1. DL vs. Traditional ML: DL models, particularly LSTM and CNN, outperform traditional ML models due to their ability to capture long-term dependencies and contextual relationships in text.
2. Balanced data: Data balancing is a crucial process for detecting fake news.
3. Effectiveness of Ensemble Learning: The HardVoting approach combining SVM, DT, and NV achieved better results than individual classifiers, confirming that ensemble methods enhance detection performance**.**
4. Hybrid Approaches and Explainability: The combination of FastText embeddings with LSTM improved fake news classification, especially in low-resource languages like Kurdish. The Hybrid FastText + Explainable AI model further demonstrated that AI-based interpretability enhances trust and usability.
5. Language-Based Performance Differences: The balanced FastText + LSTM model performed better for Kurdish (94.25%) than English (92.11%), suggesting that Kurdish text may contain stronger linguistic markers for fake news classification**.**

5.2. Limitations and Future Work

Recent advances in Transformer-based architectures, such as BERT, RoBERTa, and XLM-RoBERTa, have demonstrated superior performance on NLP tasks [4], [9] and [13]. Integrating such models for multilingual fake news detection, particularly in low-resource languages like Kurdish, remains a promising direction for future work.

- Dataset Constraints: Some models were evaluated on small datasets, which might affect generalizability.
- Language-Specific Challenges: The effectiveness of word embeddings varies across languages, requiring further improvements in morphologically rich languages like Kurdish.
- Computational Costs: DL models require high computational resources, which can be a limitation for real-time applications.

Future research could explore transformer-based architectures (e.g., BERT, XLM-RoBERTa) and multilingual FND techniques to improve performance across diverse datasets and languages.

6. Conclusion

This research evaluated the performance of various ML and DL models in detecting fake news, with a focus on Kurdish and English. It was found that balanced data and certain programs, particularly those known as LSTM and a combination of FastText and LSTM, performed significantly better than older programs. The FastText and LSTM combo scored a 94.25% accuracy rate for Kurdish and 92.11% for English. This study proves it's good at sorting out fake news even when there isn't much language resource available. But there are still some issues to resolve. For example, there isn't enough data available, some languages are more complex than others, and it costs a lot to run these programs. Future work will focus on exploring the use of transformer-based programs (such as BERT and XLM-RoBERTa) or methods capable of detecting fake news in multiple languages to improve performance across various datasets. This research gives a strong start to catching fake news in languages that don't have many resources. It also points out the benefits of using DL and mixed methods to classify pieces of text.

- Funding Sources: None.
- Conflict of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.
- Competing Interests: The Authors declare that there are no competing interests.
- Ethical and informed consent for data used: Not applicable.
- Data availability and access: The corpus data can be accessed at [15]. The implementation code will be made available at: [https://github.com/azadmk/New-Fake-News-Detection/tree/main] for reproducibility.

7. References

[1] D. A. Salh and R. M. Nabi, "Kurdish Fake News Detection Based on Machine Learning Approaches," Passer J. Basic Appl. Sci., vol. 5, no. 2, pp. 262–271, 2023, doi: 10.24271/PSR.2023.380132.1226.

[2] R. Azad, B. Mohammed, R. Mahmud, L. Zrar, and S. Sdiq, "Fake News Detection in low-resourced languages 'Kurdish language' using Machine learning algorithms," Turkish J. Comput. Math. Educ. 4219 Res. Artic1., vol. 12, no. 6, pp. 4219–4225, 2021

[3] S. Bussa, A. Bodhankar, V. H. Patil, H. Pal, S. K. Bunkar, and A. R. K. Qureshi, "An Implementation of Machine Learning Algorithm for Fake News Detection," Int. J. Recent Innov. Trends Comput. Commun., vol. 11, no. 3 (March), pp. 392–401, 2023, doi: 10.17762/ijritcc.v11i9s.7435.

[4] T. Saravanan, T. Jhaideep, and N. H. Bindu, "Detecting depression using Hybrid models created using Google's BERT and Facebook's Fast Text Algorithms," 2022 2nd Int. Conf. Adv. Comput. Innov. Technol. Eng. ICACITE 2022, no. September, pp. 415–421, 2022, doi: 10.1109/ICACITE53722.2022.9823581.

[5] R. A. M. San Ahmed, "Hard Voting Approach using SVM, Naïve Bays and Decision Tree for Kurdish Fake News Detection," Iraqi J. Comput. Sci. Math., vol. 4, no. 3, pp. 25–33, 2023, doi: 10.52866/ijcsm.2023.02.03.003.

[6] N. Sheikh, S. Parveen, A. Rehman, A. Naeem, M. Anjum, and M. Yasin, "Comparative Study of Fake News

Detection using Naïve Bayes and Logistic Model," vol. 4883, no. July, pp. 1008–1015, 2024, doi: 10.53555/ks.v12i5.3390.
[7] S. Ahmadi, "KLPT – Kurdish Language Processing Toolkit," pp. 72–84, 2020, doi: 10.18653/v1/2020.nlposs-1.11.
[8] Y. Taher, A. Moussaoui, and F. Moussaoui, "Automatic Fake News Detection based on Deep Learning, FastText and News Title," Int. J. Adv. Comput. Sci. Appl., vol. 13, no. 1, pp. 146–158, 2022, doi: 10.14569/IJACSA.2022.0130118.
[9] L. Hu, S. Wei, Z. Zhao, and B. Wu, "Deep learning for fake news detection: A comprehensive survey," AI Open, vol. 3, no. September, pp. 133–155, 2022, doi: 10.1016/j.aiopen.2022.09.001.
[10] S. Sadiq, T. Aljrees, and S. Ullah, "Deepfake Detection on Social Media: Leveraging Deep Learning and FastText Embeddings for Identifying Machine-Generated Tweets," IEEE Access, vol. 11, no. September, pp. 95008–95021, 2023, doi: 10.1109/ACCESS.2023.3308515.
[11] N. N. Prachi, M. Habibullah, M. E. H. Rafi, E. Alam, and R. Khan, "Detection of Fake News Using Machine Learning and Natural Language Processing Algorithms," J. Adv. Inf. Technol., vol. 13, no. 6, pp. 652–661, 2022, doi: 10.12720/jait.13.6.652-661.
[12] T. S. Camelia and F. R. Fahim, "A Regularized LSTM Method for Detecting Fake News Articles," IEEE, pp. 1–2, 2024.
[13] E. Hashmi, S. Y. Yayilgan, M. M. Yamin, S. Ali, and M. Abomhara, "Advancing Fake News Detection: Hybrid Deep Learning With FastText and Explainable AI," IEEE Access, vol. 12, no. March, pp. 44462–44480, 2024, doi: 10.1109/ACCESS.2024.3381038.
[14] H. Abdul, H. Al, H. Abdul, H. Al, and M. Jabardi, "Detecting Fake News Using Machine Learning: A Comparative Study of Techniques," J. Kufa Math. Comput., vol. 11, pp. 113–120, 2024.
[15] Abubakr Salh, Dana; Nabi, Rebwar (2022), "Kurdish Dataset for Fake News Detection (KDFND)", Mendeley Data, V1, doi: 10.17632/3zx9vpw3wh.1
[16] M. O. Arowolo, S. Misra, and R. O. Ogundokun, "A Machine Learning Technique for Detection of Social Media Fake News," IJSWIS, vol. 19, no. 1, pp. 1–25, 2023, doi: 10.4018/IJSWIS.326120.
[17] T. A. Rashid, A. M. Mustafa, and A. M. Saeed, "A Robust Categorization System for Kurdish Sorani Text Documents," Inf. Technol. J., vol. 16, no. 1, pp. 27–34, 2016, doi: 10.3923/itj.2017.27.34.
[18] I. Science, E. Don, S. Engineering, D. Bosco, and S. I. Science, "Deceptive Content Detection Using Machine Learning," IJSREM, pp. 1–5, 2024, doi: 10.55041/IJSREM34830.
[19] K. Yan, "Optimizing an English text reading recommendation model by integrating collaborative filtering algorithm and FastText classification method," Heliyon, vol. 10, no. 9, p. e30413, 2024, doi: 10.1016/j.heliyon.2024.e30413.