**Phase I**

**Institution:** Government College Of Technology, Coimbatore

**Department:** Computer Science and Engineering

**Date of Submission:** 25/04/2025

# Problem Statement

**“Exposing the truth with advanced fake news detection powered by natural language processing”**

The widespread circulation of fake news on digital platforms is a growing real-world problem, leading to misinformation, public confusion and social unrest. Detecting fake news manually is slow and ineffective. This issue is important to solve because timely and identification of fake news can help protect public trust, promote informed decisions and ensure the integrity of information in society.

# Objectives of the Project

# This project aims to develop an NLP-based detection system to combat the spread of fake news. The objective is to build a machine learning model that classifies news content as real or fake based on textual data. The system will provide credibility predictions with confidence scores, identify misinformation patterns and help users make informed decisions. Key outcomes include enhanced media literacy, faster detection of misleading content and a reliable solution tested on a dataset. The focus will be on providing a proof-of-concept rather than immediate deployment across digital platforms.

# Scope of the Project

This project will deliver a proof-of-concept NLP-based fake news detection system powered by an LSTM classifier.

It encompasses the following components:

**Dataset:**

* Utilize a publicly available, labeled news corpus (e.g. Kaggle “Fake and

Real News” dataset)

**Data Preprocessing:**

* Text cleaning (HTML removal, punctuation stripping)
* Tokenization, stop-word removal
* Stemming or lemmatization for normalization

**Feature Extraction:**

* TF-IDF vectorization as a baseline
* Pretrained word embeddings (Word2Vec, GloVe)
* N-gram analysis to capture phrase-level patterns

**Model Development:**

* Design and train an LSTM network with an embedding layer, dense

layers and sigmoid output

* Apply regularization (dropout, L2) to mitigate overfitting

**Evaluation:**

* Split data into training, validation and test sets
* Measure accuracy, precision, recall, F1-score and ROC-AUC
* Provide confidence scores for individual predictions

**LIMITATIONS AND CONSTRAINTS:**

* Multilingual support will not be included in the current phase; regional

and international language detection will be explored in future work.

* Integration with social media platforms such as Twitter, Facebook or

WhatsApp is not covered due to API access and policy restrictions.

# Data Sources

For this project, we will use the “Fake News Detection” dataset uploaded by Sameer Patel on Kaggle.

* **Source**: [Kaggle - Fake News Detection Dataset by Sameer Patel](https://www.kaggle.com/datasets/sameepv2/fake-news)
* **Access Type**: Public and freely available for download.
* **Nature**: The dataset is static – it is downloaded once and does not update in real-time.
* **Contents**: It contains labeled news articles with attributes like:
* title: headline of the news article
* text: full body of the news article
* label: classification (1 = fake, 0 = real)

This dataset provides a solid foundation for building and evaluating fake news detection models using NLP techniques.

# High-Level Methodology

**1. Data Collection**

* Source: Publicly available fake news datasets from Kaggle.
* Type: Labeled data containing news headlines, body text, and class labels (real/fake).
* Method: Download the dataset directly from Kaggle.

**2. Data Cleaning**

* Remove unwanted characters, punctuation, and HTML tags.
* Handle missing values by removing or imputing incomplete rows.
* Eliminate duplicate entries to prevent data leakage.
* Normalize text by converting to lowercase and fixing inconsistent formats.

**3. Exploratory Data Analysis (EDA)**

* Use bar plots, word clouds and histograms to visualize the distribution of fake vs. real news.
* Analyze word frequencies, text lengths and label balance.
* Detect common keywords or patterns in fake and real news.

**4. Feature Engineering**

* Apply TF-IDF or Word Embeddings (like GloVe or Word2Vec) to vectorize the text.
* Experiment with n-grams (uni-, bi-, tri-grams) to capture context.
* Use lemmatization/stemming and consider POS tagging or sentiment scores as features.

**5. Model Building**

* Begin with traditional models like Logistic Regression, Naive Bayes and Random Forest for benchmarking.
* Build a deep learning model using LSTM to capture the sequential nature of text.
* Embedding Layer for word vectors.
* LSTM Layers to model word sequences.
* Dense and Output layers for final classification.

**6. Model Evaluation**

* Split data into training, validation and test sets.
* Use metrics such as:
* Accuracy – overall correctness
* Precision & Recall – relevance of results
* F1-score – balance between precision and recall
* ROC-AUC – for evaluating binary classification performance
* Visualize performance using a confusion matrix.

**7. Visualization & Interpretation**

* Display results using matplotlib, seaborn, or Plotly for clarity.
* Show model predictions, performance charts, and sample classifications.
* Interpret how the model makes decisions using important words/features.

# Tools and Technologies

* **Programming Language:** Python 3.8+
* **NLP Libraries:** NLTK, spaCy, Hugging Face Transformers
* **Data Handling:** Pandas, NumPy
* **Visualization:** Matplotlib, Seaborn
* **Machine Learning:** Scikit-learn
* **Deep Learning:** TensorFlow (with Keras) or PyTorch
* **Text Vectorization:** TF-IDF, Word2Vec, GloVe
* **Model Architecture:** LSTM (with Embedding Layer, Dense Layers, Sigmoid Output)
* **Dataset Source:** Kaggle (Fake News Dataset)

# Team Members and Roles

**Bhavadharani M**

* Co-leads data collection and cleaning.
* Writes sections of the final project report.
* Participates in model evaluation and visualization.

**Dhanya P**

* Manages the GitHub repo and version control.
* Contributes to model training.
* Coordinates team meetings and milestone check-ins.

**Gayathri S**

* Co-develops ML/NLP models.
* Helps with project documentation and dataset description.
* Helps create evaluation metrics and results tables.

**Rinchin Pelton**

* Works on text preprocessing and tokenization.
* Integrates modules from all team members.
* Assists in writing code explanations and documentation.

**Varshini R**

* Assists in hyperparameter tuning and training scripts.
* Leads testing and validation (cross-validation, confusion matrix, etc.).
* Contributes to performance analysis and interprets model outputs.