Project FakeScope: Intelligent Fake News Detector

Table of Contents

Project Proposal

Problem Statement / Product Vision

Challenge: The proliferation of fake news has negatively impacted public opinion and decision-making across social, political, and economic spheres. Addressing this phenomenon requires automated, accurate tools to assess the veracity of information published online.

Extra-Challenge: Since fake news are now a global problem, to provide a multi-lingual version, at least in Spanish, and English.

Product Vision: FakeScope aims to create an intelligent system capable of automatically detecting fake news in digital press articles and providing, alongside each evaluation, a quantitative credibility score (0-100) and a brief automatic justification, easy to just look up. The solution will combine natural language processing (NLP), automated fact-checking via external APIs, robust semantic evaluation, and automatic explanation generation.

Initial Hypotheses:

- It is possible to significantly improve automatic fake news detection by combining modern NLP pipelines and external fact-checking models.

- Using semantic comparison techniques and cross-verification with external sources provides more accurate and transparent results than a closed model.
- Using the recollected data for doing deep insight about the state of the media: *is this journal reliable? In election period, should I trust this editorial line?*

Data Collection

- **Base Data:** Use of labeled datasets such as "Fake News Detection" (Kaggle, Huggingface, Datasetsearch Google) and other public repositories of real and fake news.
  - **Main Challenge For Scalability**: to compile the data from users in order to get a better score from models and refine the database.
- **External Fact-checking:** Use of APIs such as Google Fact Check Tools, Gemini FactChecker AI, and multimedia verification resources (InVID) to cross-check text claims.
- **Preprocessing:** Normalisation, tokenisation, and lemmatisation to adapt data to the processing and detection pipeline.

References / Literature

- Google Fact Check Tools API - https://developers.google.com/fact-check/tools/api?hl=es-419
- Fake News Detection using Machine Learning : https://www.geeksforgeeks.org/machine-learning/fake-news-detection-using-machine-learning/
- Automated credibility assessment (GitHub - credibilityScore) Implementation returning credibility scores 0-10 based on journalistic standards: formality, neutrality, transparency, and layout. : https://github.com/Lzdnl/credibilityScore
- "Combining Semantics and Context for Improved Fake News Detection" (Yin et al., 2022)
- Language-Specific versus Multilingual Models for Fact-Checking (Bernal-Beltrán et al., 2025) : https://ceur-ws.org/Vol-4038/paper_64.pdf
- The perils and promises of fact-checking with large language models (Quelle et al., 2024)

Architecture Diagram (Flow)

The FakeScope system follows this processing pipeline:

1. **Input:** News article
2. **Text Preprocessing:** cleaning, normalization, embedding
3. **Key Claims Extraction (NLP)**

4. **External Fact-checking (APIs: Google, Gemini, InVID)**
5. **Semantic Comparison with Verified Sources**
6. **Credibility Score (0-100)**
7. **Automatic Comment Generation (T5/FLAN-T5)**
8. **Dashboard Visualisation (Streamlit)**

High-level Repository Architecture

Main Directories and Components

- **/data**: Datasets, including raw and processed CSV/Parquet files
- **/notebooks**: Jupyter/Colab notebooks for EDA, modeling and evaluation
- **/src**: Core Python source code
  - **/preprocessing**: Tokenization, lemmatization, and embedding scripts
  - **/models**: Training scripts, model definitions (logistic regression, BERT, etc.)
  - **/fact_checking**: API clients (Google, Gemini, InVID) and cross-verification logic
  - **/explanation**: Explanation/comment generator (T5/FLAN)
  - **/dashboard**: Streamlit dashboard app
- **/tests**: Unit and integration tests
- **/scripts**: Auxiliary scripts (data download, preprocessing automation)
- **README.md**: General project documentation
- **requirements.txt / environment.yml**: Python dependencies

High-level Organization

```
FakeScope/
├── data/
├── notebooks/
├── src/
│   ├── preprocessing/
│   ├── models/
│   ├── fact_checking/
│   ├── explanation/
│   └── dashboard/
├── tests/
├── scripts/
├── README.md
└── requirements.txt | environment.yml
```

Trello Board: Main Project Steps

Phase 1: Data & Preprocessing

- Download labeled fake news datasets (Kaggle, etc.)
- Clean and preprocess data (tokenization, lemmatization)
- Explore data for insights (EDA notebook)
- Generate TF-IDF and embedding vectors
- Save processed datasets (CSV/Parquet)

Phase 2: Modeling & Fact-checking

- Train baseline models (LogReg, RandomForest, Naive Bayes)
- Train advanced models (BERT, DistilBERT, RoBERTa)
- Integrate Google Fact Check API
- Integrate Gemini FactChecker AI
- Extract key claims with NLP
- Implement semantic similarity comparison
- Evaluate models (metrics: accuracy, precision, recall, F1)

Phase 3: Generation & Results Visualization

- Develop dashboard (Streamlit)
- Generate explanatory comments through T5/FLAN
- Visualize results: credibility score, reliability graph, links to sources
- Test on new articles
- Create final project report and documentation

Phase 4: Deployment & Maintenance

- Finalize and clean code repo
- Add end-to-end tests and documentation
- Prepare for deployment (requirements, config)