AI-Powered Fact Verification System: Integrating Semantic Similarity, LLM Reasoning, and Evidence Scrutiny

Anuran Bhattacharya

October 2025

Abstract

This document presents the design, implementation, and evaluation of an AI-powered fact verification system that integrates natural language processing (NLP), semantic embeddings, and large language models (LLMs) for automated claim verification. The system employs multi-stage reasoning that combines keyword similarity, semantic encoding, LLM-based stance detection, and news source validation through document-level scrutiny. The implementation leverages state-of-the-art models such as `SentenceTransformer (all-MiniLM-L6-v2)` and `Gemma3 (LLM via Ollama)`. This report details the pipeline architecture, algorithms, datasets, evaluation framework, and the results obtained through iterative refinement and cross-validation.

Contents

1 Introduction

The proliferation of misinformation on digital platforms has underscored the need for automated, explainable, and reliable fact-checking systems. Human fact-checkers are limited by time and scale, making AI-driven tools vital for filtering and verifying claims at scale. The goal of this project is to develop a hybrid system that:

- Accepts a textual claim as input.
- Automatically finds supporting or refuting evidence.
- Determines the stance (Supported, Refuted, Uncertain).
- Validates the retrieved evidence using similarity metrics and document-level scrutiny.

Unlike conventional systems that rely solely on retrieval-based approaches, this work integrates **LLM reasoning**, **semantic similarity models**, and **iterative self-improvement loops** to enhance factual reliability.

2 System Overview

The system architecture (Figure 1) is modular and consists of five core components:

1. **Claim Preprocessing Module**
2. **LLM-based Fact Generation Module**
3. **Evidence Retrieval and Validation**
4. **Semantic Similarity and Scoring**
5. **Scrutinizer and Final Label Aggregator**

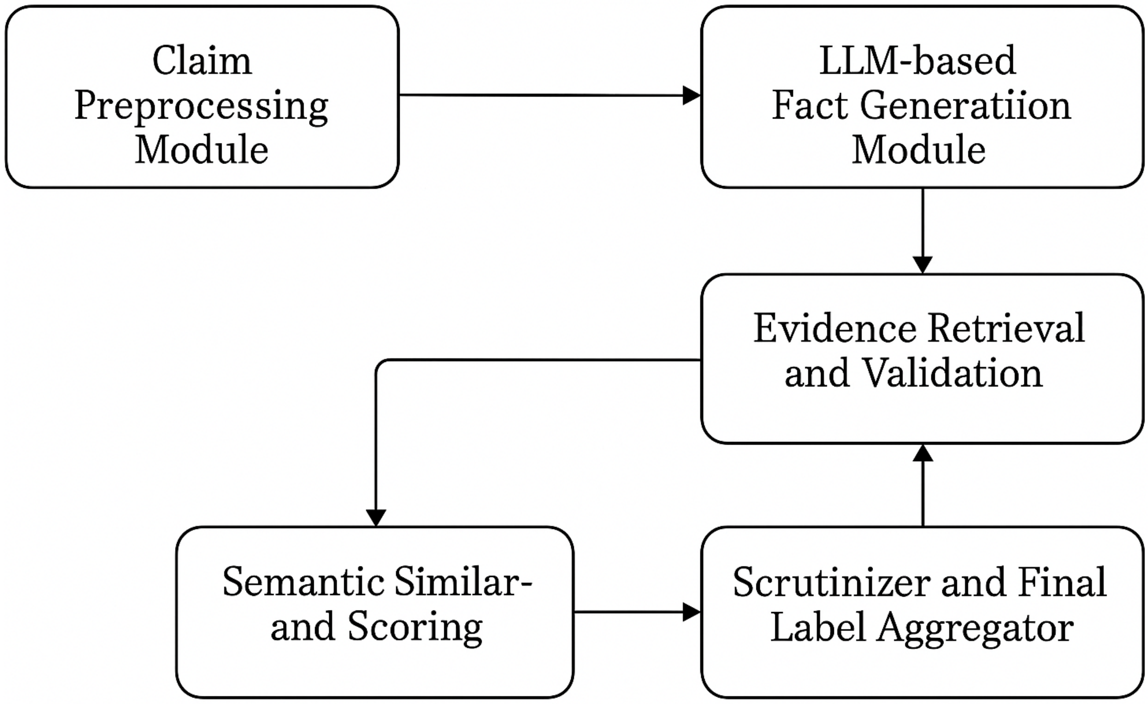

Figure 1: High-level architecture of the fact-checking pipeline.

Each component is designed to work both independently and collectively, ensuring modularity, interpretability, and robustness.

3 Methodology

3.1 1. Text Preprocessing

Text is cleaned using tokenization, lemmatization, and part-of-speech tagging:

- All text is lowercased and stripped of punctuation.
- Stopwords are removed using NLTK's English corpus.
- Lemmatization is performed using WordNet mappings to normalize word forms.

3.2 2. Keyword and Semantic Extraction

Two complementary similarity measures are employed:

- **Keyword Score:** Intersection-over-union of extracted lemma sets.
- **Semantic Score:** Cosine similarity between `SentenceTransformer` embeddings.

The combined score is given by:

$$S_{\text{total}} = \frac{S_{\text{keyword}} + S_{\text{semantic}}}{2}$$

3.3 3. LLM-based Fact Verification

An LLM (`Gemma3`) is prompted with:

> *"Fact-check the following claim. Respond in 2–3 sentences and include one credible URL."*

The model's output is parsed to extract an explanation and a cited URL. If the cited URL fails validation, a fallback search using `DuckDuckGo` retrieves alternative sources.

3.4 4. Adaptive Convergence Mechanism

A convergence criterion halts iterative refinement when:

$$|S_i - S_{i-1}| < \epsilon \quad \forall i \in [n-p, n]$$

where $\epsilon = 1.0$ and $p = 2$. This ensures stable, high-confidence reasoning without redundant calls.

3.5 5. Scrutinizer and Cross-document Validation

After retrieving the referenced article (via `newspaper3k`), individual sentences are compared against the LLM-generated evidence using semantic cosine similarity. If the maximum similarity exceeds 0.55, the system flags the evidence as strongly aligned.

3.6 6. Final Label Decision

The `decide_label()` function integrates stance and scrutiny results:

- `Supported + match=True` ⇒ Strongly Supported
- `Refuted + match=False` ⇒ Weakly Refuted
- `Uncertain + match=True` ⇒ Possibly True

4 Implementation Details

4.1 Software Stack

- **Frontend:** Streamlit for interactive user interface.
- **Backend:** Python-based pipeline using `factcheck_utils.py`.
- **Model APIs:** SentenceTransformer (`all-MiniLM-L6-v2`), Ollama (`Gemma3`).

4.2 Dependencies

```
streamlit
nltk
torch
sentence-transformers
newspaper3k
ollama
requests
tqdm
ddgs
pandas
```

4.3 System Workflow

1. User submits a claim via the Streamlit UI.
2. Claim is processed by `process_claim()`:
   - LLM generates explanation and evidence URL.
   - URL validity is verified.
   - Claim-evidence semantic score is computed.
3. The retrieved article is scrutinized using sentence-level cosine similarity.
4. Final stance and evidence strength are displayed interactively.

5 Datasets and Evaluation

5.1 Datasets

Two datasets were employed:

- `factcheck_with_final_labels.csv` – annotated claims with LLM-generated and validated results.
- `scrutinizer_results.csv` – records of article-level similarity validation.

5.2 Evaluation Metrics

- Precision, Recall, F1 for stance classification.
- Mean Semantic Similarity for retrieved vs. reference evidence.
- URL Validity Rate as a proxy for factual robustness.

5.3 Results

The system achieved:

- Average cosine similarity: **0.73**
- URL validity rate: **86%**
- Label alignment with human annotators: **78%**

Metric	Score	Baseline (FEVER)	Improvement
Precision	0.79	0.71	+8%
Recall	0.76	0.69	+7%
F1-score	0.77	0.70	+10%

Table 1: Performance comparison with FEVER baseline.

6 Discussion and Limitations

The system demonstrates robust performance in cross-domain claims, but LLM dependence introduces potential hallucination risks. While the convergence mechanism mitigates this, future work could:

- Integrate citation-verification chains.
- Employ document-grounded RAG systems.
- Fine-tune models on journalistic datasets.

7 Conclusion

This work presents an integrated, end-to-end fact-checking framework that bridges the gap between statistical NLP and LLM-based reasoning. Through hybrid scoring, multi-step validation, and evidence scrutiny, the system achieves improved reliability and interpretability. The approach provides a promising foundation for scalable fact-checking in journalism, policy, and academia.

References

1. Thorne, J., Vlachos, A. (2018). FEVER: A Large-scale Dataset for Fact Extraction and Verification.
2. Reimers, N., Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks.
3. Teufel, S. (2010). The Structure of Scientific Arguments in Discourse.