Misinformation Detection System: A Novel NLP-Based Approach

March 2, 2025

Abstract

This paper presents a novel methodology for misinformation detection in news articles using Large Language Models (LLMs), search engines, and evidence extraction techniques. The system verifies claims extracted from news articles by searching reputable sources and assigning a misinformation score. Additionally, we compare this approach with a traditional NLP-based method that evaluates sentiment, readability, textual similarity, and source reliability. Our novel approach is dynamic, scalable, and context-aware, while the secondary approach provides supplementary insights into misinformation trends.

1 Introduction

Misinformation has become a global issue, with fake news influencing public opinion and decision-making. Traditional methods for detecting misinformation rely on NLP metrics such as sentiment analysis and readability scoring, which provide indirect indicators of misinformation. However, they often lack contextual verification against authoritative sources.

Our novel approach directly extracts factual claims, searches for corroborating evidence from reputable sources, and assigns a *misinformation score* using an LLM-based reasoning system. This method leverages:

- Claim extraction from news articles using Google's Gemini LLM,
- Google search to find supporting or contradicting evidence,
- Large Language Models (LLMs) for misinformation assessment.

We also incorporate a secondary NLP-based approach to supplement analysis, combining sentiment analysis, readability scores, and textual similarity to evaluate article credibility.

2 Novel Approach: Claim-Based Misinformation Detection

2.1 Extracting Factual Claims

We utilize Google's Gemini LLM to extract key factual claims from a given news article. This ensures that only verifiable statements are analyzed:

Listing 1: Claim Extraction Using Gemini

```
def extract_claims_from_url(url):
    response = requests.get(url, timeout=5)
    soup = BeautifulSoup(response.text, "html.parser")
    paragraphs = soup.find_all("p")
    text = " ".join(p.get_text() for p in paragraphs)[:3000]

    prompt = f"Extract key factual claims from this article:\n\n{text}"
    claims_response = llm_gemini.invoke(prompt)

    if claims_response and claims_response.content:
```

```
        return [c.strip() for c in claims_response.content.split("\n") if c.strip()]
    return []
```

2.2 Searching for Evidence

Each extracted claim is verified using Google Custom Search, restricted to reputable news sources. The query formulation optimizes for high-quality evidence retrieval:

Listing 2: Google Search for Evidence

```
def search_google(query):
    news_filter = " OR ".join([f"site:{site}" for site in NEWS_SITES])
    query = f"{query} {news_filter}"

    url = "https://www.googleapis.com/customsearch/v1"
    params = {"q": query, "key": GOOGLE_API_KEY, "cx": GOOGLE_CSE_ID, "num": 5}

    response = requests.get(url, params=params, timeout=10)
    return [r["link"] for r in response.json().get("items", [])]
```

2.3 Assigning a Misinformation Score

The extracted claim and retrieved evidence are fed into the Gemini LLM, which assigns a *misinformation score* (0–100). The scale is defined as:

$$MS = f(C, E)$$

where:

- C = Extracted Claim
- E = Evidence Retrieved

Listing 3: Misinformation Score Calculation

```
prompt = PromptTemplate(
    input_variables=["claim", "evidence"],
    template="""
    Claim: {claim}
    Evidence: {evidence}

    Assign a misinformation score (0-100) based on credibility, contradictions, and verification:
    "Misinformation Score: X/100 - [Short reason]"
    """
)

response = llm_gemini.invoke(prompt.format(claim=claim, evidence=evidence))
```

3 Secondary Approach: NLP-Based News Evaluation

While our novel claim-based approach directly verifies factual accuracy, we also implement a supplementary NLP-based evaluation method to provide additional insights.

3.1 Source Reliability Score

We use a predefined list of unreliable news domains to check whether a news article comes from a credible source:

$$S_{\text{source}} = \begin{cases} 1.0 & \text{if domain is reliable,} \\ 0.0 & \text{if domain is unreliable.} \end{cases}$$

Listing 4: Source Reliability Check

```
def check_source_reliability(domain, unreliable_list):
    return 0.0 if domain.lower() in unreliable_list else 1.0
```

3.2 Sentiment Analysis

Sentiment analysis is performed using VADER, where:

$$S_{\text{sentiment}} = \frac{P - N}{P + N + \text{NEU}}$$

Listing 5: Sentiment Analysis

```
def get_sentiment_score(text):
    analyzer = SentimentIntensityAnalyzer()
    scores = analyzer.polarity_scores(text)
    return (scores['pos'] - scores['neg']) / (scores['pos'] + scores['neg'] +
        scores['neu'])
```

3.3 Readability Score

We use the Flesch Reading Ease metric:

$$RE = 206.835 \; - \; 1.015 \left(\frac{W}{S} \right) \; - \; 84.6 \left(\frac{\text{SYL}}{W} \right)$$

3.4 Overall Score Calculation

A weighted aggregation of NLP-based metrics is computed as:

$$S_{\text{final}} = \frac{\sum w_i S_i}{\sum w_i}$$

Listing 6: Overall Score Computation

```
def compute_overall_score(source_score, sentiment, readability, rouge, cosine_sim,
    tfidf, bias):
    metrics = {
        "source_score": (source_score, 0.3),
        "sentiment": (1 - abs(sentiment), 0.1),
        "readability": (min(max(readability / 100, 0), 1), 0.1),
        "rouge": (rouge, 0.1),
        "cosine_sim": (cosine_sim, 0.1),
        "tfidf": (tfidf, 0.15),
        "bias": (1 - bias, 0.15)
    }
    return sum(val * wgt for val, wgt in metrics.values()) / sum(wgt for val, wgt
        in metrics.values())
```

4 Conclusion

Our claim-based misinformation detection approach provides a fact-centered, scalable, and high-accuracy method by verifying claims against reputable sources. The NLP-based secondary approach offers additional context, ensuring a comprehensive evaluation of news articles.

Future improvements may integrate real-time AI-driven fact-checking, multilingual support, and blockchain-based credibility tracking to further enhance the misinformation detection landscape.