

A Comparative Analysis of Large Language Models for Factual Claim Extraction and Verification

submitted by

Aasem Elshahat

Matriculation Number: 954435

to Department VI - Media Informatics and Media
Berliner Hochschule für Technik

to obtain the academic degree

Bachelor of Science (B.Sc.)

in **Medieninformatik**

Date of Submission: January 2, 2026

Supervisors:

Prof. Dr. Siamak Haschemi

Dipl.-Inf. (FH) Markus Schubert

Examiner:

Prof. Dr. Felix Gers



Abstract

Large Language Models (LLMs) have transformed natural language processing, yet their tendency to generate factually incorrect information, known as hallucination, poses significant risks for applications requiring factual accuracy. This thesis presents a comparative analysis of three state-of-the-art LLMs (gpt-4o-mini, gemini-2.5-flash, and deepseek-v3.2) within TruthLens, a custom implementation of a decoupled two-phase architecture adapted for multi-model benchmarking.

Evaluating performance on a benchmark derived from the BingCheck dataset, we identified distinct behavioral patterns across the models. For claim extraction, gpt-4o-mini achieves superior F1-Score (80.4%) through a balanced precision-recall strategy, while gemini-2.5-flash and deepseek-v3.2 adopt conservative approaches with high precision but significantly reduced recall. For verification, models achieve 77–82% accuracy on straightforward claims but show a marked performance drop on minority classes (Refuted and Insufficient Information). This discrepancy highlights the challenge of distinguishing between false information and missing evidence in imbalanced datasets.

Qualitative analysis highlights three key failure modes: the tendency to reject context-dependent claims, the generation of ineffective search queries, and the models' struggles with minority classes in imbalanced datasets. Despite these challenges, models achieve 69% unanimous correctness on verification tasks, demonstrating that LLM-based fact-checking has reached practical reliability for straightforward factual claims. This work contributes a reusable multi-model framework, empirical performance benchmarks, and actionable guidance for real-world deployment.

Zusammenfassung

Large Language Models (LLMs) haben die Verarbeitung natürlicher Sprache grundlegend verändert, doch ihre Tendenz, faktisch falsche Informationen zu generieren – bekannt als Halluzination – birgt erhebliche Risiken für Anwendungen, die faktische Genauigkeit erfordern. Diese Arbeit präsentiert eine vergleichende Analyse von drei hochmodernen LLMs (gpt-4o-mini, gemini-2.5-flash und deepseek-v3.2) innerhalb von TruthLens, einer eigens entwickelten Implementierung einer entkoppelten Zwei-Phasen-Architektur für Multi-Modell-Benchmarking.

Bei der Evaluierung anhand eines aus dem BingCheck-Datensatz abgeleiteten Benchmarks wurden unterschiedliche Verhaltensmuster der Modelle identifiziert. Bei der Behauptungsextraktion erzielt gpt-4o-mini einen überlegenen F1-Score (80,4%) durch eine ausgewogene Precision-Recall-Strategie, während gemini-2.5-flash und deepseek-v3.2 konservative Ansätze mit hoher Precision, aber deutlich reduziertem Recall verfolgen. Bei der Verifikation erreichen die Modelle 77–82% Genauigkeit bei eindeutigen Behauptungen, zeigen jedoch einen deutlichen Leistungsabfall bei Minderheitsklassen (Widerlegt und Unzureichende Informationen). Diese Diskrepanz verdeutlicht die Herausforderung, in unbalancierten Datensätzen zwischen falschen Informationen und fehlenden Belegen zu unterscheiden.

Die qualitative Analyse hebt drei wesentliche Fehlermodi hervor: die Tendenz, kontextabhängige Behauptungen abzulehnen, die Generierung ineffektiver Suchanfragen sowie die Schwierigkeiten der Modelle mit Minderheitsklassen in unbalancierten Datensätzen. Trotz dieser Herausforderungen erreichen die Modelle bei 69% der Verifikationsaufgaben einstimmige Korrektheit, was zeigt, dass LLM-basierte Faktenprüfung für eindeutige faktische Behauptungen eine praktische Zuverlässigkeit erreicht hat. Diese Arbeit liefert ein wiederverwendbares Multi-Modell-Framework, empirische Leistungsbenchmarks sowie praxisorientierte Empfehlungen für den realen Einsatz.

Acknowledgements

This section will be completed in the final version of the thesis.

Contents

List of Abbreviations	IX
1 Introduction	1
1.1 Motivation	1
1.1.1 The Problem of LLM Hallucinations	1
1.1.2 The Need for Automated Fact-Checking	2
1.2 Objective and Research Questions	2
1.2.1 Thesis Objective	2
1.2.2 Research Questions	3
1.3 Structure of the Thesis	3
2 Background and State of the Art	5
2.1 Large Language Models and Factuality	5
2.2 Phase 1: Factual Claim Extraction	6
2.2.1 Literature Review: The Atomic Claim Approach	6
2.2.2 The BingCheck Dataset	7
2.3 Phase 2: Factual Claim Verification	8
2.3.1 The Knowledge Cutoff Problem	8
2.3.2 Literature Review: Search-Augmented Verification	8
2.4 The TruthLens Framework	9
2.4.1 Foundation: The ClaimeAI Architecture	9
2.4.2 Scope of Adaptation	9
3 Methodology	11
3.1 Experimental Framework	11
3.1.1 Experimental Design and Variables	12
3.1.2 LLM Providers	13
3.2 Dataset Creation and Sampling	13
3.2.1 Source Dataset	13
3.2.2 Sampling Strategy (Phase 1)	13
3.2.3 Ground Truth Generation	13

3.3	Decoupled Two-Phase Evaluation	14
3.3.1	Phase 1: Claim Extraction (The "Recall" Task)	14
3.3.2	Phase 2: Claim Verification (The "Precision" Task)	14
3.4	Evaluation Metrics	15
3.4.1	Phase 1: Claim Extraction Metrics (Binary)	15
3.4.2	Phase 2: Claim Verification Metrics (Multi-Class)	16
3.5	Reliability and Reproducibility	16
3.6	Threats to Validity	17
3.6.1	Internal Validity	17
3.6.2	External Validity	17
3.6.3	Construct Validity	17
4	Implementation	19
4.1	System Architecture	19
4.1.1	Agent Workflows	20
4.2	Provider Management Infrastructure	22
4.2.1	Provider Interface	22
4.2.2	DeepSeek Structured Output Middleware	23
4.2.3	Search Provider Extension	23
4.3	Benchmark Automation	24
4.3.1	Phase 1: Extraction Automation	24
4.3.2	Phase 2: Verification Automation	25
4.3.3	Data Storage and Serialization	25
4.4	Analysis Pipeline	26
4.4.1	Phase 1 Analysis (Extraction)	26
4.4.2	Phase 2 Analysis (Verification)	26
5	Results and Evaluation	27
5.1	Phase 1: Claim Extraction Performance	27
5.1.1	Quantitative Metric Comparison	27
5.1.2	Negative Class Performance	28
5.1.3	Visual Analysis: Precision-Recall Trade-off	28
5.1.4	Confusion Matrix Analysis	29
5.1.5	Reliability Analysis Across Multiple Runs	30
5.2	Phase 2: Claim Verification Performance	30
5.2.1	Quantitative Metric Comparison	30
5.2.2	Impact of Class Imbalance	31
5.2.3	Visual Comparison	32
5.2.4	Inter-Model Agreement Analysis	32
5.2.5	Reliability Analysis Across Multiple Runs	33

5.3	Summary of Key Findings	34
6	Discussion	36
6.1	Comparative Analysis of LLMs	36
6.1.1	Best Performing LLM for Extraction	36
6.1.2	Best Performing LLM for Verification	37
6.1.3	The Stability-Performance Trade-off	37
6.2	Qualitative Error Analysis	38
6.2.1	Common Failure Patterns in Claim Extraction	38
6.2.2	Common Failure Patterns in Claim Verification	38
6.2.3	The 69% Unanimous Correctness	39
6.3	Limitations of the Study	40
6.3.1	Sample Size and Domain Specificity	40
6.3.2	Constraints of the TruthLens Architecture	40
6.3.3	Threats to Validity	41
7	Conclusion and Future Work	42
7.1	Summary of Findings	42
7.1.1	Answering Research Question 1 (Extraction)	42
7.1.2	Answering Research Question 2 (Verification)	43
7.1.3	Answering Research Question 3 (Qualitative)	43
7.2	Contributions	44
7.3	Practical Implications	45
7.4	Future Work	45
7.4.1	Alternative Search Providers	45
7.4.2	Multi-Agent Consensus Architectures	46
7.4.3	Self-Correction Mechanisms	46
7.4.4	Adversarial Robustness	47
7.4.5	Domain-Specific and Multilingual Extensions	48
7.4.6	Real-World Deployment and User Studies	49
7.5	Closing Remarks	49
A	Appendix	51
A.1	Full Benchmark Results Tables	51
A.1.1	Phase 1: Extraction Results Per Run	51
A.1.2	Phase 2: Verification Results Per Run	51
A.1.3	Confusion Matrices	51
A.1.4	Inter-Model Agreement Analysis	52
A.2	Prompt Templates	52
A.2.1	Claim Extraction Prompts	53
A.2.2	Claim Verification Prompts	55

A.3	Model Configuration	56
A.3.1	Provider Settings	56
A.3.2	Generation Parameters	56
A.3.3	Search Configuration	57
A.3.4	Environment Variables	57
A.4	Benchmark Runner Script	57
A.5	Analysis Script	59
A.6	Dataset Sample	60
A.6.1	Phase 1: Claim Extraction Sample	60
A.6.2	Phase 2: Claim Verification Sample	62

List of Figures

3.1	Experimental Pipeline: Data flow from sampling through evaluation phases to final metrics analysis.	12
4.1	High-level architecture of the TruthLens Multi-Agent System. Reproduced from [BharathxD, 2024].	20
4.2	The Claim Extraction Workflow implementing the Claimify methodology. Reproduced from [BharathxD, 2024].	21
4.3	The Claim Verifier Workflow implementing the SAFE reasoning loop. Reproduced from [BharathxD, 2024].	22
5.1	Precision-Recall trade-off for claim extraction. gpt-4o-mini occupies the balanced region near $F1=0.8$, while gemini-2.5-flash and deepseek-v3.2 cluster in the high-precision, low-recall quadrant.	29
5.2	Confusion matrices for claim extraction (averaged across $k = 3$ runs). gpt-4o-mini shows balanced errors, while gemini-2.5-flash and deepseek-v3.2 exhibit high false negative counts (47–39 missed claims).	29
5.3	Run-to-run variability for extraction phase ($k = 3$ runs). gpt-4o-mini demonstrates the most consistent performance with minimal variance.	30
5.4	Per-class F1-scores for verification. The heatmap reveals consistently poor performance on minority classes (Refuted: $n = 3$, Insufficient: $n = 12$) compared to the majority class (Supported: $n = 85$).	31
5.5	Verification performance comparison. The gap between Accuracy and Macro-F1 indicates that high accuracy is driven primarily by majority class performance.	32
5.6	Inter-model agreement rates for verification verdicts. Fleiss' $\kappa = 0.41$ indicates moderate agreement across all three models.	33
5.7	Run-to-run variability for verification phase ($k = 2$ runs). gpt-4o-mini produces identical results across runs, while deepseek-v3.2 shows the highest variability.	34

5.8	Radar chart comparing model performance across all key metrics. gpt-4o-mini demonstrates the most balanced profile, while gemini-2.5-flash and deepseek-v3.2 show specialized strengths.	35
-----	--	----

List of Tables

1.1	Specifications of the Large Language Models evaluated in this study. .	3
2.1	Classic classification of Hallucinations as defined by [Ji et al., 2023]. .	5
2.2	Example entry from the BingCheck dataset showing sentence-level annotations	7
4.1	Extraction Dataset Schema. The [model] prefix is replaced with provider names (e.g., <code>gpt4_binary_result</code>).	24
4.2	Verification Benchmark Schema. The [model] prefix is replaced with provider names (e.g., <code>gpt4_verdict</code> , <code>gemini_reasoning</code>).	25
5.1	Phase 1 Extraction Metrics (Positive Class). Mean \pm standard deviation across $k = 3$ runs. $N = 150$ sentences (78 positive, 72 negative).	27
5.2	Phase 1 Extraction Metrics (Negative Class). Specificity measures the ability to correctly identify non-factual content; NPV (Negative Predictive Value) indicates reliability when predicting “no claim.” . .	28
5.3	Phase 2 Verification Metrics. Mean \pm standard deviation across $k = 2$ runs. $N = 100$ claims (85 Supported, 3 Refuted, 12 Insufficient). . . .	31
5.4	Summary of model performance across both experimental phases. Bold indicates best performance for each criterion.	35
A.1	Claim Extraction Metrics Per Run ($N = 150$ sentences)	51
A.2	Claim Verification Metrics Per Run ($N = 100$ claims)	52
A.3	Averaged Confusion Matrices for Claim Extraction ($k = 3$ runs) . . .	52
A.4	Inter-Model Agreement and Error Analysis (Verification Phase) . . .	52
A.5	LLM API Configuration (exact model identifiers used in code)	56
A.6	LLM Generation and network parameters (as implemented in code) .	56
A.7	Sample Sentences from Experimental Dataset (BingCheck-derived) . .	61
A.8	Extraction Phase Sample: Argentine Tango	61
A.9	Verification Phase Sample: Argentine Tango	62

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
CAG	Credibility-aware Generation
CoVe	Chain-of-Verification
CSV	Comma-Separated Values
FN	False Negative
FP	False Positive
JSON	JavaScript Object Notation
LLM	Large Language Model
NLG	Natural Language Generation
NLP	Natural Language Processing
NPV	Negative Predictive Value
RAG	Retrieval-Augmented Generation
RQ	Research Question
SAFE	Search-Augmented Factuality Evaluation
SD	Standard Deviation
TN	True Negative
TP	True Positive
URL	Uniform Resource Locator
UX	User Experience

Chapter 1

Introduction

The advent of Large Language Models (LLMs) has fundamentally transformed the landscape of Natural Language Processing (NLP) and information retrieval. Models such as OpenAI’s gpt-4o-mini [OpenAI, 2024], Google’s gemini-2.5-flash [Gemini Team, Google, 2025], and DeepSeek’s deepseek-v3.2 [DeepSeek-AI, 2025] have demonstrated unprecedented capabilities in generating human-like text, summarizing complex documents, and answering open-domain questions. However, the widespread adoption of these systems in critical domains—such as journalism, legal research, and education—is severely hindered by their tendency to generate plausible but factually incorrect information, a phenomenon widely known as “hallucination” [Ji et al., 2023].

As LLMs are increasingly integrated into search engines, the ability to automatically verify the factual veracity of their outputs has become a paramount research challenge. This thesis addresses this challenge by developing TruthLens, a custom implementation of a decoupled two-phase architecture adapted from the open-source *ClaiameAI* repository [BharathxD, 2024]¹, for automated fact-checking and to benchmark the performance of state-of-the-art LLMs (gpt-4o-mini, gemini-2.5-flash, deepseek-v3.2) within this system.

1.1 Motivation

1.1.1 The Problem of LLM Hallucinations

While LLMs are fluent and coherent, they lack an inherent understanding of truth. They operate as probabilistic systems, predicting the next token (the fundamental unit of text processing [Jurafsky and Martin, 2025]) based on statistical patterns

¹Original repository available at: <https://github.com/BharathxD/ClaiameAI> (Accessed: 2025-12-22)

learned during training rather than querying a structured knowledge base of verified facts. Consequently, these models often conflate correct information with outdated data, common misconceptions, or complete fabrications [Ji et al., 2023].

The risk is not merely academic; in high-stakes applications, hallucinations can lead to the propagation of misinformation, legal liabilities, and the erosion of user trust. For instance, an LLM might generate a summary of a news article that accurately captures the tone but invents specific dates or statistics. This “fluency-factuality gap”—where the text reads perfectly but is factually wrong—makes manual verification difficult and time-consuming for human users.

1.1.2 The Need for Automated Fact-Checking

Manual fact-checking is unscalable given the volume of content generated by LLMs. Traditional fact-checking pipelines often rely on human annotators or static knowledge graphs, which cannot keep pace with the dynamic nature of information on the web [Thorne et al., 2018].

Recent approaches have proposed using LLMs themselves as evaluators (LLM-as-a-Judge) [Zheng et al., 2023]. However, existing evaluation frameworks often treat fact-checking as a monolithic task, asking a model to “verify this text” in a single step. This approach obscures the root cause of errors: did the model fail to notice a verifiable claim (extraction error), or did it fail to verify the claim against evidence (verification error)?

To address this, there is a need for a granular, agent-based evaluation framework that decouples *extraction* from *verification*. By isolating these phases, researchers can identify whether a model is better suited for the high-recall task of identifying claims or the high-precision task of judging them against ground truth.

1.2 Objective and Research Questions

1.2.1 Thesis Objective

The primary objective of this thesis is to perform a comparative analysis of three leading Large Language Models—**gpt-4o-mini**, **gemini-2.5-flash**, and **deepseek-v3.2**—to determine their efficacy in an automated fact-checking pipeline.

Unlike previous studies that evaluate end-to-end performance, this research adapts and extends the open-source *ClaiMeAI* architecture into a system referred to as **TruthLens**. This custom benchmarking pipeline enables the comparative evaluation of the *BingCheck* dataset [Li et al., 2024] to assess each model’s performance in:

Provider	Model Name	Version / Endpoint
OpenAI	gpt-4o-mini	openai:gpt-4o-mini
Google	gemini-2.5-flash	gemini-2.5-flash
DeepSeek	deepseek-v3.2	deepseek-chat (V3.2 API)

Table 1.1: Specifications of the Large Language Models evaluated in this study.

1. **Phase 1 (Extraction):** Identifying verifiable factual claims from unstructured sentences [Metropolitansky and Larson, 2025].
2. **Phase 2 (Verification):** Determining the accuracy of the LLMs in assessing the veracity of those claims against a standardized ground truth, utilizing a search-augmented verification approach [Wei et al., 2024].

This thesis aims to establish which model offers the optimal balance between precision (trustworthiness) and recall (coverage) for building reliable automated fact-checking agents.

1.2.2 Research Questions

To achieve the stated objective, this thesis answers the following three research questions (RQs):

RQ1: Claim Extraction Which LLM most accurately identifies verifiable factual claims from text at the sentence level?

Specifically, how do the models compare in terms of Recall (avoiding missed claims) versus Precision (avoiding subjective noise)?

RQ2: Claim Verification Which LLM most accurately assesses the veracity of identified claims when provided with claims?

RQ3: Qualitative Analysis What are the common failure modes in LLM-based fact-checking?

1.3 Structure of the Thesis

The remainder of this thesis is structured as follows:

Chapter 2 provides the theoretical background on Large Language Models, the phenomenon of hallucination, and the state-of-the-art in automated fact-checking, including a review of the SAFE framework [Wei et al., 2024] and the BingCheck dataset.

Chapter 3 details the experimental methodology. It defines the TruthLens framework, the decoupled evaluation strategy, and the specific definitions of metrics such

as F1-Score and atomic claims used throughout the study.

Chapter 4 describes the technical implementation of the evaluation pipeline, including the Python-based agent architecture, the integration of LLM APIs, and the automated benchmarking scripts.

Chapter 5 presents the quantitative results of the experiments. It offers a statistical comparison of gpt-4o-mini, gemini-2.5-flash, and deepseek-v3.2 across both extraction and verification phases, analyzing accuracy, reliability, and inter-model agreement across multiple experimental runs.

Chapter 6 interprets the findings, discussing the implications of the “precision-recall trade-off” observed in the models. It also provides a qualitative error analysis of specific failure cases.

Chapter 7 summarizes the contributions of this work and suggests future directions for agent-based fact-checking systems.

Chapter 2

Background and State of the Art

This chapter establishes the theoretical foundations of the thesis. It reviews the phenomenon of hallucination in Large Language Models (LLMs), analyzes the state-of-the-art methodologies for factual claim extraction and verification, and introduces the theoretical basis for the TruthLens framework.

2.1 Large Language Models and Factuality

The capabilities of Large Language Models (LLMs) have scaled dramatically in recent years, driven by advancements in transformer architectures [Vaswani et al., 2017] and the expansion of pre-training datasets. However, these models remain prone to “hallucination”—the generation of text that is fluent and coherent but factually incorrect [Ji et al., 2023].

Early research in Natural Language Generation (NLG) primarily categorized these errors relative to a source document (e.g., in summarization tasks). This produced the standard distinction between **Intrinsic** and **Extrinsic** hallucinations (Table 2.1).

Type	Definition	Example Scenario
Intrinsic Hallucination	The generated output directly contradicts the source material or input context.	Input: “The cat is black.” Output: “The cat is white.”
Extrinsic Hallucination	The generated output cannot be verified from the source (neither supported nor contradicted), effectively constituting a fabrication.	Input: “The cat is black.” Output: “The cat is named Felix.” (Source does not contain this information).

Table 2.1: Classic classification of Hallucinations as defined by [Ji et al., 2023].

However, as LLMs have evolved into open-ended agents that rely on world knowledge rather than just a single input document, recent literature argues for a refined taxonomy. In this context, an *agent* is defined as an autonomous system that utilizes an LLM as its central controller to perceive tasks, formulate execution plans, and interact with external tools (such as search engines) to achieve specific goals [Wang et al., 2024]. Recent surveys [Huang et al., 2024] distinguish between two dominant failure modes in modern Generative AI:

1. **Faithfulness Hallucination:** The model fails to follow user instructions or maintains internal inconsistency (e.g., self-contradiction).
2. **Factuality Hallucination:** The model generates content that contradicts verifiable real-world facts. This manifests as *Factual Fabrication* (inventing non-existent entities) or *Factual Contradiction* (misstating established relations).

This thesis focuses specifically on detecting and mitigating **Factuality Hallucinations**. Unlike faithfulness errors, which can often be solved via prompt engineering, factuality errors require external verification pipelines—such as the TruthLens framework proposed in this study—to audit the model’s output against reliable ground truth.

2.2 Phase 1: Factual Claim Extraction

The first step in any fact-checking pipeline is identifying *what* needs to be checked. Raw LLM output often interweaves verifiable facts with opinions, reasoning, and conversational filler. Evaluating the text as a monolithic block often leads to imprecise results.

2.2.1 Literature Review: The Atomic Claim Approach

State-of-the-art research suggests that effective evaluation requires breaking down long-form text into "atomic claims." An atomic claim is defined as a specific, verifiable proposition that is fully self-contained [Metropolitansky and Larson, 2025].

Recent work by Metropolitansky and Larson [Metropolitansky and Larson, 2025] introduced **Claimify**, a framework that formalized this process. They identified that naive sentence splitting is insufficient because it strips necessary context (e.g., resolving pronouns like "he" or "it"). Their research proposed a multi-stage pipeline consisting of:

1. **Selection:** Filtering out subjective or non-verifiable sentences.
2. **Disambiguation:** Resolving coreferences (e.g., changing "He published the

theory" to "Albert Einstein published the theory") to make sentences self-contained.

3. **Decomposition:** Breaking complex compound sentences into individual atomic facts.

This thesis adopts the Claimify methodology for Phase 1, positing that higher-quality extraction leads to more accurate verification downstream.

2.2.2 The BingCheck Dataset

To quantitatively evaluate the performance of automated systems, a rigorous **ground truth** is required. In the context of natural language processing, ground truth refers to a set of data labeled by human experts that serves as the reference standard of correctness against which model outputs are measured. Without this reference standard, it is impossible to calculate objective performance metrics such as Precision (how many predicted claims are valid) and Recall (how many valid claims were found).

This study leverages the **BingCheck** corpus [Li et al., 2024], a high-quality dataset originally designed for evaluating Retrieval-Augmented Generation (RAG) systems across diverse domains such as Science, History, and Finance. The dataset provides a granular breakdown of generated responses, mapping unique Answer IDs to specific questions and segmenting the response into individual sentences with binary factuality labels.

Table 2.2 illustrates the exact structure of the dataset using two sequential rows from the corpus. It demonstrates how non-factual conversational filler (Sentence 0) is distinguished from verifiable factual assertions (Sentence 1) within the same response context.

Answer ID	Question	Sentence ID	Sentence	Label
ea262d2b...	What is the history and cultural significance behind the traditional Argentine tango...?	0	That’s a great question.	False
ea262d2b...		1	The Argentine tango is a dance and musical form that originated in Buenos Aires...	True

Table 2.2: Representative rows from the BingCheck dataset. The *Label* column serves as the ground truth for Phase 1, where *False* indicates no factual claim and *True* indicates a verifiable claim. (Note: Answer IDs are truncated for display).

While the original BingCheck dataset focuses on document-level verification, this thesis utilizes these sentence-level annotations to derive the binary classification

ground truth for Phase 1 (Extraction).

2.3 Phase 2: Factual Claim Verification

Once atomic claims are extracted, they must be verified against external evidence. As comprehensively surveyed by Guo et al. [Guo et al., 2022], automated fact-checking has evolved from early rule-based systems to sophisticated neural approaches that integrate evidence retrieval with veracity prediction. Relying on an LLM’s internal parametric knowledge (facts encoded in model weights during training) is insufficient due to the “knowledge cutoff” problem and the risk of reinforcing existing hallucinations.

2.3.1 The Knowledge Cutoff Problem

Large Language Models are trained on static datasets with fixed temporal boundaries, meaning their parametric knowledge becomes progressively outdated after training completion. Cheng et al. [Cheng et al., 2024] systematically analyzed this phenomenon, demonstrating that LLMs exhibit measurable degradation in factual accuracy for events occurring after their training cutoff dates. For the models evaluated in this thesis, these cutoffs range from late 2023 to mid-2025, rendering them unreliable for verifying claims about recent events, updated statistics, or evolving scientific consensus. This limitation fundamentally motivates the search-augmented approach adopted in the TruthLens framework.

2.3.2 Literature Review: Search-Augmented Verification

The prevailing standard for automated verification is **Search-Augmented Factuality Evaluation (SAFE)**, proposed by researchers at Google DeepMind, Stanford University, and University of Illinois at Urbana-Champaign [Wei et al., 2024]. SAFE introduces the paradigm of using an LLM agent to interact with a search engine (e.g., Google Search) to verify claims.

The SAFE framework operates on a "reasoning loop":

1. **Query Generation:** The model formulates search queries based on the claim.
2. **Evidence Retrieval:** Search results are parsed to find supporting or refuting evidence.
3. **Reasoning:** The model compares the claim against the retrieved evidence to issue a verdict (Supported, Irrelevant, or Not Supported).

The authors of the SAFE framework [Wei et al., 2024] demonstrated that LLM agents using this method can achieve human-level performance in rating long-form factuality, significantly outperforming human crowdsourced annotators in terms of cost and scalability. This thesis adapts the SAFE logic into the verification agent of the TruthLens framework.

2.4 The TruthLens Framework

To empirically evaluate the performance of gpt-4o-mini, gemini-2.5-flash, and deepseek-v3.2, this thesis implements **TruthLens**. TruthLens is a specialized adaptation and extension of the open-source *ClaimeAI* framework, with significant architectural extensions to support systematic comparative multi-model benchmarking.

2.4.1 Foundation: The ClaimeAI Architecture

The architectural foundation of this study is **ClaimeAI** [BharathxD, 2024], an open-source fact-checking tool designed by BharathxD. ClaimeAI introduced the concept of a decoupled pipeline utilizing **LangGraph** [LangChain AI, 2024]. Unlike traditional linear chains, LangGraph is a library specifically designed for building stateful, multi-actor applications with LLMs. It enables the creation of cyclical graphs, allowing agents to loop, persist state, and make iterative decisions—a critical capability for the complex reasoning required in fact-checking tasks. Within this framework, the process is split into two autonomous agents::

1. **Claim Extractor:** Uses a specific graph of prompt nodes (Selection \rightarrow Disambiguation \rightarrow Decomposition) to isolate atomic claims.
2. **Claim Verifier:** Uses a search-augmented loop (Search \rightarrow Reason \rightarrow Verdict) to verify those claims.

However, the original ClaimeAI repository was configured to use a single LLM provider (OpenAI GPT-4) with the model identifier hardcoded in the configuration. While the underlying LangChain framework (via `init_chat_model`) theoretically supports multiple providers, the original implementation lacked the systematic infrastructure required for controlled comparative benchmarking across multiple LLM providers with different API capabilities.

2.4.2 Scope of Adaptation

The original ClaimeAI repository was configured for single-provider execution with OpenAI GPT-4 hardcoded in the configuration. To conduct the multi-model comparative analysis required by this thesis, systematic extensions were necessary. These

extensions address four key requirements: (1) ensuring experimental consistency across providers with different API capabilities, (2) enabling large-scale batch processing for statistical validation, (3) providing cost-effective search infrastructure, and (4) restoring the three-label verification schema. Chapter 3 details the methodological rationale for these requirements, and Chapter 4 describes their technical implementation.

Chapter 3

Methodology

This chapter details the experimental framework designed to answer the research questions. It defines the specific Large Language Models (LLMs) selected for comparison, the structure of the TruthLens agentic framework, and the rigorous metric definitions used to evaluate performance in both extraction and verification phases.

3.1 Experimental Framework

To perform a fair and reproducible comparative analysis, this study utilizes **TruthLens**—an experimental adaptation of the open-source *ClaimeAI* repository [BharathxD, 2024].¹

While the original repository was configured for single-model execution (OpenAI GPT-4), this study required systematic infrastructure for controlled comparative benchmarking. Four methodological requirements drove the design of TruthLens:

1. **Experimental Consistency Across Providers:** A fair comparison requires that all models receive identical treatment. However, the three target providers (OpenAI, Google, DeepSeek) have different API capabilities: OpenAI supports native structured output, Gemini requires explicit safety configuration to prevent false refusals on factual news content, and DeepSeek lacks native schema validation. Without a unified abstraction layer, these asymmetries would introduce confounding variables—observed differences might reflect implementation artifacts rather than model capabilities. The TruthLens design therefore mandates that all models pass through identical configuration logic, validation, and error handling.

¹Original repository available at: <https://github.com/BharathxD/ClaimeAI> (Accessed: 2025-12-22)

2. **Large-Scale Batch Processing:** The original ClaimeAI was designed for interactive, one-claim-at-a-time processing. However, rigorous benchmarking requires processing $N = 150$ sentences across 3 models with $k = 3$ runs each—a total of 1,350 extraction API calls, plus 600 verification calls. This scale necessitates automated batch processing with rate limiting (to respect API quotas), failure recovery (to handle transient network errors), and result serialization (to preserve data for analysis). Additionally, checkpointing mechanisms prevent data loss during long-running experiments.
3. **Cost-Effective Search Infrastructure:** The original ClaimeAI relied on expensive neural search providers (Tavily, Exa) unsuitable for thesis-scale experimentation. To enable the required experimental runs within budget constraints, support for the Brave Search API [Brave Software, 2025] was added as a cost-effective, keyword-based retrieval backend. The design abstracts different search backends behind a unified interface, ensuring that the choice of search provider does not affect agent logic.
4. **Three-Label Verification Schema:** The original ClaimeAI had disabled support for *Insufficient Information* verdicts, limiting verification to binary classification (*Supported* vs. *Refuted*). However, real-world fact-checking often encounters claims where evidence is genuinely inconclusive. Restoring the three-label schema (*Supported*, *Refuted*, *Insufficient Information*) enables more realistic evaluation and aligns with standard fact-checking taxonomies used in datasets like BingCheck.

The technical implementation of these requirements is detailed in Chapter 4.

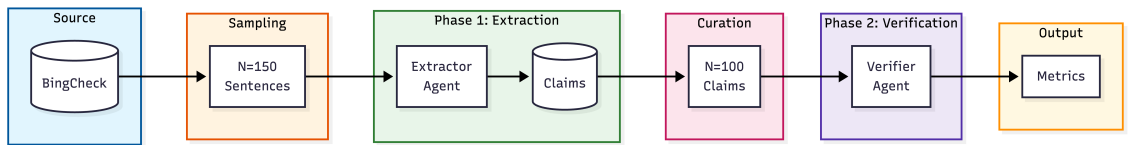


Figure 3.1: Experimental Pipeline: Data flow from sampling through evaluation phases to final metrics analysis.

3.1.1 Experimental Design and Variables

The experiment is designed as a comparative study where the **Independent Variable** is the Large Language Model used as the cognitive engine for the agents. The **Dependent Variables** are the performance metrics (Precision, Recall, F1-Score, and Accuracy) achieved by each model.

To ensure internal validity, the following controls were strictly enforced:

- **Identical Prompts:** All three models received the exact same system instructions and few-shot examples (task demonstrations included in the prompt [Jurafsky and Martin, 2025]) for both extraction and verification tasks. No model-specific prompt engineering was applied.
- **Deterministic Sampling:** To minimize variability in generation, the temperature settings (a hyperparameter controlling output randomness [Jurafsky and Martin, 2025]) were standardized (where applicable) to maximize output reproducibility.

3.1.2 LLM Providers

The study benchmarks three models representing different architectural philosophies (Proprietary vs. Open Weights [Open Source Initiative, 2024]) and cost profiles. Open Weights models publicly release their model parameters, enabling local deployment and fine-tuning, while proprietary models are accessible only through vendor APIs. The specific versions used for all experiments are listed in Table 1.1 (see Chapter 1).

3.2 Dataset Creation and Sampling

3.2.1 Source Dataset

The source text for this experiment is derived from the **BingCheck** dataset [Li et al., 2024], the characteristics and structure of which are detailed in Section 2.2.2.

3.2.2 Sampling Strategy (Phase 1)

Due to the computational intensity of agentic workflows, a stratified random sample was utilized rather than processing the entire corpus. Using a fixed random seed (`random_state=42`) to ensure the sampling process is deterministic and reproducible, $N = 150$ sentences were sampled from the source corpus.

3.2.3 Ground Truth Generation

A critical challenge in automated fact-checking is establishing a reliable "Gold Standard." This study adopts a hybrid approach:

- **Phase 1 (Extraction):** The ground truth for extraction was derived from the expert annotations published by the authors of the *Claimify* framework [Metropolitansky and Larson, 2025]. Each of the 150 sampled sentences possesses a binary label (*True/False*) indicating whether it contains a factual claim.

- **Phase 2 (Verification):** For the verification phase, a standardized benchmark of 100 atomic claims was required. The experimental procedure was as follows: (1) Phase 1 extraction was executed with all three models, (2) gpt-4o-mini was identified as the best-performing extractor (F1-Score = 80.4%), and (3) the first 100 claims from gpt-4o-mini’s extraction output were selected as the Phase 2 benchmark dataset. This methodological choice ensures that all verifiers are evaluated on identical, well-formed atomic claims, prioritizing benchmark quality and reproducibility. While this introduces a potential limitation regarding claim characteristic generalization (discussed in Section 3.6), it reflects the practical scenario where production systems would deploy the optimal extractor. The ground truth was established via manual human annotation by the author using a **three-label schema**:

- **Supported:** The claim is explicitly confirmed by authoritative sources.
- **Refuted:** The claim is explicitly contradicted by authoritative sources.
- **Insufficient Information:** No reliable evidence could be found to either support or refute the claim (e.g., unverifiable private information or ambiguous predictions), or the claim itself was too ambiguous or unclear to be verified (e.g., ambiguous subjects).

This alignment ensures that if a model correctly identifies a claim as unverifiable, it is penalized not as an error but rewarded as a correct assessment of uncertainty.

3.3 Decoupled Two-Phase Evaluation

The TruthLens framework executes the evaluation in two distinct, sequential phases.

3.3.1 Phase 1: Claim Extraction (The "Recall" Task)

The objective of Phase 1 is to convert unstructured input text into a list of *atomic claims*. An atomic claim must be verifiable and decontextualized. For this phase, the `claim_extractor` agent processes the 150 sampled sentences. The output is evaluated as a binary classification task: for every sentence in the ground truth, did the model correctly identify it as containing a claim?

3.3.2 Phase 2: Claim Verification (The "Precision" Task)

The objective of Phase 2 is to verify the accuracy of the claims. The `claim_verifier` agent utilizes a search-augmented loop (Search → Read → Reason) to compare the

claim against web evidence. The output is a multi-class classification: *Supported*, *Refuted*, or *Insufficient Information*. The model’s verdict is compared against the manually annotated ground truth to calculate accuracy.

3.4 Evaluation Metrics

Reliable measurement requires distinct metrics for each phase.

3.4.1 Phase 1: Claim Extraction Metrics (Binary)

In Phase 1, the task is treated as a binary classification problem where the **Positive Class** represents the presence of a verifiable claim, and the **Negative Class** represents its absence (e.g., subjective opinions or conversational filler).

While the Claimify study [Metropolitansky and Larson, 2025] propose linguistic metrics such as Entailment and Decontextualization coverage, this study prioritizes Binary Recall as the primary success metric. In a high-throughput fact-checking pipeline, the critical failure mode is missing a claim entirely (False Negative). Therefore, we treat extraction as a detection task (Found/Not Found) rather than a generation quality task. To fully capture the Precision-Recall trade-off (RQ1), metrics are calculated for both classes.

Positive Class Metrics (Target: Factual Claims)

These metrics evaluate the model’s ability to correctly identify factual content:

- **Precision (+):** $\frac{TP}{TP+FP}$

Interpretation: Resistance to Noise. Of the sentences the LLM classified as "factual," how many actually were? High precision implies the model avoids hallucinating claims in subjective text.

- **Recall (+):** $\frac{TP}{TP+FN}$

Interpretation: Coverage. Of the sentences that actually contained facts, how many did the LLM find?

- **F1-Score (+):** $2 \times \frac{Precision(+)\times Recall(+)}{Precision(+)+Recall(+)}$

The harmonic mean of Precision (+) and Recall (+). This is the primary ranking metric for extraction performance.

Negative Class Metrics (Target: Non-Factual Content)

These metrics evaluate the model’s ability to correctly reject non-factual content:

- **Negative Predictive Value (NPV):** $\frac{TN}{TN+FN}$

Interpretation: When the model predicts a sentence is non-factual, how often is it correct?

- **Specificity / Recall (-):** $\frac{TN}{TN+FP}$

Interpretation: **Filtering Capability.** Of the subjective/non-factual sentences in the dataset, how many did the model correctly ignore? This directly measures the model’s ability to filter out subjective noise.

- **F1-Score (-):** $2 \times \frac{NPV \times Specificity}{NPV + Specificity}$

Interpretation: The harmonic mean for the negative class. A high score here indicates a balanced ability to filter noise without aggressively rejecting valid facts.

3.4.2 Phase 2: Claim Verification Metrics (Multi-Class)

In Phase 2, the model must classify a claim into one of three categories: *Supported*, *Refuted*, or *Insufficient Information*.

The primary metric is **Accuracy**, calculated as the ratio of correct verdicts to total claims:

$$Accuracy = \frac{\text{Correct Verdicts}}{\text{Total Claims Verified}} \quad (3.1)$$

Additionally, we calculate the **Macro-Averaged F1-Score** across all three classes. This ensures that the model is evaluated not just on its ability to verify facts, but also on its ability to correctly identify when evidence is missing (Recall for the “Insufficient” class), preventing bias towards the majority class.

3.5 Reliability and Reproducibility

Given that Large Language Models are non-deterministic by nature, a single experimental run may not accurately reflect the model’s stable performance characteristics. To mitigate this variability and ensure statistical reliability, the evaluation pipeline is executed across **multiple independent runs**.

Due to the computational intensity and API costs associated with the verification phase (which requires multiple search queries per claim), the number of runs differs between phases:

- **Phase 1 (Extraction):** $k = 3$ independent runs
- **Phase 2 (Verification):** $k = 2$ independent runs

For each metric M (Precision, Recall, F1-Score), the reported result is the mean across all runs:

$$M_{final} = \frac{1}{k} \sum_{i=1}^k M_i \quad (3.2)$$

This multi-run approach allows for the calculation of standard deviation to quantify the stability of each model’s extraction and verification capabilities.

3.6 Threats to Validity

This section acknowledges the limitations that may affect the generalizability and interpretation of results.

3.6.1 Internal Validity

- **LLM Non-Determinism:** Large Language Models produce variable outputs even with identical inputs. This threat is mitigated by executing $k = 3$ independent runs and reporting mean performance with standard deviation.
- **API Version Drift:** Cloud-hosted models may be updated by providers during the experimental period. All experiments were conducted within a concentrated timeframe (December 2025) to minimize this effect.

3.6.2 External Validity

- **Dataset Scope:** The BingCheck dataset, while spanning diverse domains such as Science, History, and Finance, represents a specific distribution of factual claims. Results may not generalize to highly specialized domains (e.g., advanced medical or legal claims).
- **Sample Size:** The evaluation uses $N = 150$ sentences for extraction and $N = 100$ claims for verification. While sufficient for comparative analysis, larger-scale studies may reveal different performance patterns.

3.6.3 Construct Validity

- **Ground Truth Subjectivity:** The Phase 2 ground truth relies on manual annotation, which may introduce subjective bias. This threat was mitigated through two measures: (1) adhering to a standardized three-label schema with unambiguous definitions for each verdict category (Supported, Refuted, Insufficient), and (2) documenting the authoritative source URL for each annotation in the dataset, enabling full traceability and independent verification of the ground truth labels.

- **Search Engine Variability:** The Brave Search API may return different results over time as web content changes, potentially affecting verification reproducibility.
- **Verification Benchmark Composition:** The Phase 2 verification dataset consists of claims extracted by the best-performing extraction model (gpt-4o-mini, F1=80.4%). This methodological decision represents a fundamental trade-off between *benchmark quality* and *input diversity*. Using high-quality, well-formed claims ensures that the verification evaluation measures each model’s core capability to assess factuality against evidence, rather than confounding the results by testing robustness to extraction errors (e.g., malformed or ambiguous claims). While the verifier receives intermediate extraction outputs (claim text, disambiguated sentence, original sentence), these fields contain no model-specific reasoning or confidence scores that could advantage any particular verification model—all three verifiers process identical inputs. This approach assumes that claim characteristics (e.g., complexity, specificity, domain coverage) are similar across extractors for claims they successfully identify. The design reflects realistic production scenarios where systems would deploy the optimal extractor, making the benchmark representative of practical usage. Alternative approaches (e.g., using claims from all extractors or human-annotated claims) would introduce different trade-offs: testing verifier robustness to poor extractions versus requiring significantly higher manual annotation costs. Due to budget constraints limiting the study to $N = 100$ manually annotated claims, the chosen approach prioritizes fair comparison of verification capability under controlled conditions.

Chapter 4

Implementation

This chapter details the technical realization of the TruthLens framework. It describes the system architecture, the engineering of the provider management infrastructure for systematic multi-model benchmarking, and the development of the automated benchmarking pipeline used to execute the experiments.

4.1 System Architecture

The TruthLens framework is built upon a modular, multi-agent architecture designed to decouple the task of claim extraction from claim verification. The system is implemented in Python using **LangGraph** [LangChain AI, 2024], a library for building stateful, multi-actor applications with Large Language Models (LLMs).

The initial prototype (based on the open-source *ClaimAI* repository) provided a baseline implementation configured for single-provider execution. To address the methodological requirements outlined in Section 3.1, the architecture was extended to support provider management, batch processing with statistical validation, and automated metric calculation.

The high-level architecture consists of three primary components:

1. **The Agent Core:** Two specialized autonomous agents (Extractor and Verifier) that execute the logic defined in the *Claimify* and *SAFE* methodologies—adapted from the original ClaimAI framework.
2. **The Provider Management Layer:** A unified interface implementing the Strategy Pattern that standardizes interactions across providers with different API capabilities and configuration requirements, ensuring experimental consistency.

3. **The Benchmarking Engine:** Automated pipeline for large-scale execution ($N = 150$ samples, $k = 3$ runs per phase, 3 models) with rate limiting, failure recovery, multi-run statistical analysis, and result serialization.

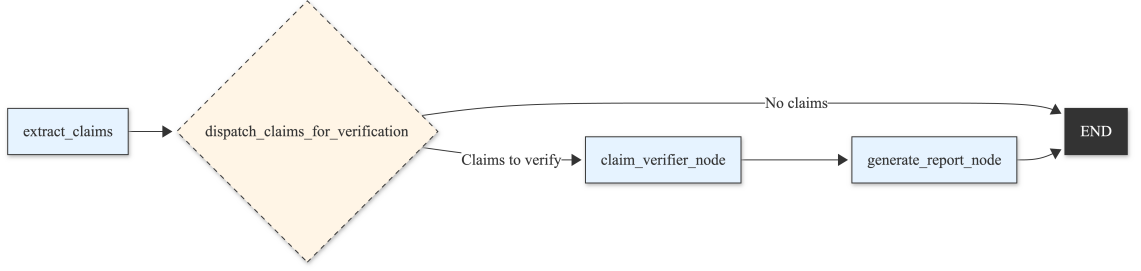


Figure 4.1: High-level architecture of the TruthLens Multi-Agent System. Reproduced from [BharathxD, 2024].

4.1.1 Agent Workflows

The system utilizes two distinct **stateful cyclic graphs** to manage the state of the fact-checking process.

The Claim Extractor Agent

The extraction pipeline is adopted directly from the ClaimeAI repository [BharathxD, 2024], which implements the Claimify methodology [Metropolitansky and Larson, 2025]. **No modifications were made to the core extraction logic**; our adaptations (detailed in Section 3.1) occur at the provider and infrastructure layers, not within the pipeline itself.

As illustrated in Figure 4.2, the agent receives raw text and processes it through five sequential nodes:

- **Sentence Splitter:** Segments text while preserving context windows (5 preceding sentences).
- **Selection Node:** Filters out non-factual sentences using a consensus voting mechanism ($k = 3$ completions) to reduce false positives.
- **Disambiguation Node:** Resolves coreferences (e.g., replacing “he” with “Albert Einstein”).
- **Decomposition Node:** Breaks complex compound sentences into atomic claims.
- **Validation Node:** Performs a final syntax check on the extracted claims.



Figure 4.2: The Claim Extraction Workflow implementing the Claimify methodology. Reproduced from [BharathxD, 2024].

The Claim Verifier Agent

The verification pipeline is also adopted from ClaimeAI [BharathxD, 2024], which implements the SAFE methodology [Wei et al., 2024]. Unlike the extractor, **one modification was made**: we restored the *Insufficient Information* verdict category, which had been disabled in the original codebase (see Section 3.1, Requirement 4).

The Claim Verifier implements an **iterative reasoning loop** designed to mimic human fact-checking behavior (Figure 4.3). Defined in `claim_verifier/agent.py`, the agent utilizes a state graph that allows it to gather evidence dynamically until sufficient information is obtained.

The workflow consists of four distinct nodes:

- **Generate Search Query:** Analyzes the atomic claim and formulates precise, neutral search queries (e.g., converting “He won the election” to “winner of 2024 US presidential election”).
- **Retrieve Evidence:** Executes the generated queries using the configured search provider and parses the results into structured evidence objects.
- **Search Decision (The “Brain”):** This is the critical control node. It evaluates the retrieved evidence against the claim to determine sufficiency. As implemented in `nodes/search_decision.py`, the model outputs a structured decision:
 - *Needs More Evidence (True)*: The agent generates a new, refined query and loops back to the retrieval step.
 - *Needs More Evidence (False)*: The agent proceeds to the final evaluation.

To prevent infinite loops, a hard limit (configurable via `max_iterations`) forces the agent to conclude after $N = 5$ attempts.

- **Evaluate Evidence:** The final node synthesizes all accumulated evidence to issue a verdict (*Supported*, *Refuted*, or *Insufficient Information*) along with a reasoning trace.

The critical logic for the *Search Decision* node is enforced via a structured Pydantic model [Colvin et al., 2024], which compels the LLM to justify its decision before

stopping. The model must explicitly identify missing aspects before requesting more evidence.

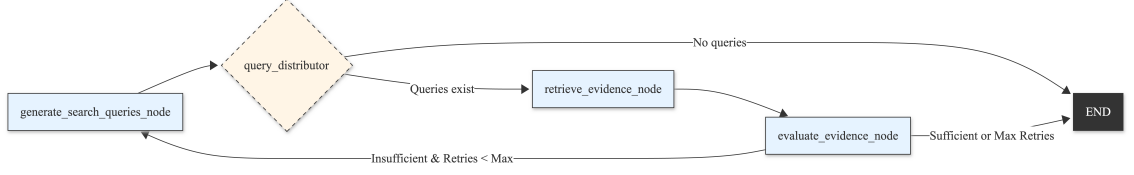


Figure 4.3: The Claim Verifier Workflow implementing the SAFE reasoning loop. Reproduced from [BharathxD, 2024].

4.2 Provider Management Infrastructure

This section describes the technical implementation of the unified provider abstraction layer required for experimental consistency (see Section 3.1 for the methodological rationale). The core challenge was reconciling the need for identical code paths across all models with the reality that each provider has different API capabilities: OpenAI supports native structured output, Gemini requires explicit safety configuration, and DeepSeek lacks schema validation.

The solution implements the Strategy Pattern [Gamma et al., 1994], encapsulating provider-specific initialization logic behind a uniform interface while ensuring all models receive identical treatment in the agent workflows.

4.2.1 Provider Interface

We defined an abstract base class, `LLMProvider`, which enforces a uniform interface for all model interactions despite underlying differences in initialization methods and API requirements. This abstraction ensures that all three models—regardless of whether they use `init_chat_model`, `ChatGoogleGenerativeAI`, or `ChatOpenAI` with custom configuration—are accessed through identical code paths in the agent logic.

```

class LLMProvider(ABC):
    @abstractmethod
    def invoke(
        self,
        model_name: str = None,
        temperature: float = 0.0,
        completions: int = 1,
    ) -> BaseChatModel:
        """Get LLM instance with specified configuration."""

```

pass

Concrete implementations were developed for three providers:

- **OpenAIProvider:** Interfaces with `gpt-4o-mini`.
- **GeminiProvider:** Interfaces with Google’s `gemini-2.5-flash`, utilizing specific safety setting overrides to prevent false refusals on sensitive news topics.
- **DeepSeekProvider:** Interfaces with `deepseek-chat` (V3.2).

4.2.2 DeepSeek Structured Output Middleware

A significant technical challenge in supporting DeepSeek models was their lack of native compatibility with LangChain’s `with_structured_output()` method. While OpenAI and Gemini support strict JSON schema validation natively, DeepSeek V3.2 only provides basic JSON mode (`response_format={"type": "json_object"}`) without schema enforcement.

To bridge this gap, we implemented a custom middleware class (`DeepSeekChatWrapper`) that provides Pydantic schema compatibility by:

1. Injecting JSON format instructions into the prompt with example schema structure
2. Parsing the raw text response to extract JSON from markdown code blocks
3. Normalizing enum field values (e.g., "SUPPORTED" → "Supported") to match expected casing
4. Validating against the target Pydantic model before returning to the agent

This adapter enables DeepSeek models to function within LangGraph’s type-safe agent framework while maintaining the same structured output guarantees as the other providers.

4.2.3 Search Provider Extension

The original repository included a search interface supporting expensive neural search APIs like Exa (formerly Metaphor) and Tavily. To enable large-scale benchmarking within the budget constraints of this study, this module was extended to support the **Brave Search API**.

The implementation in `search/provider.py` abstracts these distinct backends behind a unified asynchronous interface. This allows the agents to switch between providers without logic changes:

1. **Brave Search:** Integrated specifically for this thesis to provide cost-effective, keyword-based retrieval at scale.
2. **Exa & Tavily:** Retained from the original codebase for legacy support, though not used in the primary experiments due to cost.

To handle network instability, the search provider was wrapped with an exponential backoff retry mechanism to ensure transient failures do not crash the verification agent.

4.3 Benchmark Automation

To conduct the comparative analysis, a robust benchmarking pipeline was developed to automate the execution of experiments across the dataset.

4.3.1 Phase 1: Extraction Automation

The script `run_extraction_phase.py` manages the batch processing of the $N = 150$ sentences. It iterates through the target models (gpt-4o-mini, gemini-2.5-flash, deepseek-v3.2), dynamically injecting the appropriate `LLMProvider` into the extraction graph. Results are serialized row-by-row into a CSV file adhering to the schema in Table 4.1.

Field Name	Description
<code>answer_id</code>	Unique identifier from the BingCheck dataset.
<code>sentence</code>	The input text being analyzed for factual claims.
<code>contains_factual_claim</code>	Ground truth binary label (True/False).
<code>[model]_binary_result</code>	The model’s prediction: whether the sentence contains a claim (True/False). Repeated for each provider (gpt4, gemini, deepseek).
<code>[model]_extracted_claims_json</code>	JSON array containing the list of atomic claims extracted by the model.

Table 4.1: Extraction Dataset Schema. The `[model]` prefix is replaced with provider names (e.g., `gpt4_binary_result`).

To ensure the statistical reliability defined in the methodology ($k = 3$ runs), the automation script includes a `generate_unique_filename` utility. This ensures that subsequent experimental runs do not overwrite previous data, automatically versioning output files (e.g., `dataset_run2.csv`, `dataset_run3.csv`) to preserve a complete audit trail of all iterations.

4.3.2 Phase 2: Verification Automation

The verification phase is orchestrated by `run_verification_phase.py`. This script loads the standardized benchmark claims generated in Phase 1 and feeds them into the verification agent. The output is stored using the schema defined in Table 4.2.

Field Name	Description
<code>claim_id</code>	Unique identifier for the claim (e.g., B001_C00).
<code>claim_text</code>	The atomic proposition to be verified.
<code>original_sentence</code>	The source sentence from which the claim was extracted.
<code>ground_truth_verdict</code>	Manual annotation (Supported / Refuted / Insufficient).
<code>[model]_verdict</code>	The LLM’s final verification decision. Repeated for each provider.
<code>[model]_reasoning</code>	The model’s chain-of-thought explanation for its verdict.
<code>[model]_sources</code>	JSON array of URLs retrieved during the search-augmented verification.

Table 4.2: Verification Benchmark Schema. The `[model]` prefix is replaced with provider names (e.g., `gpt4_verdict`, `gemini_reasoning`).

Key engineering features of the verification runner include:

- **Per-Claim State Persistence:** Given that verifying a single claim may involve multiple search iterations (taking 30-60 seconds), the script saves results to the CSV after *every* single transaction, using the structure defined in Table 4.2. This “checkpointing” strategy prevents data loss in the event of API timeouts or rate limit disconnects.
- **Dynamic Schema Updates:** The script dynamically creates distinct columns for each model (e.g., `gpt4_verdict`, `gemini_verdict`, `deepseek_verdict`) within the same master dataset, facilitating direct row-by-row comparison during analysis.
- **Error Resilience:** Failed verifications (e.g., due to strict content filters on sensitive topics) are caught and logged as `None`, allowing the pipeline to continue processing remaining claims without crashing.

4.3.3 Data Storage and Serialization

The complex, nested output of the LangGraph agents (containing claim objects and metadata) is serialized and appended to the master dataset. The resulting CSV file utilizes a **wide-format structure**, where each input sentence is preserved as a row, and the extraction results for each model (gpt-4o-mini, gemini-2.5-flash, deepseek-v3.2) are stored in distinct columns. This structure facilitates direct, side-by-side comparison of model performance on identical inputs.

4.4 Analysis Pipeline

The final component of the implementation is the dual-phase analysis suite, consisting of `analyze_extraction.py` and `analyze_verification.py`. These scripts perform the statistical evaluation of the raw data against the ground truth.

4.4.1 Phase 1 Analysis (Extraction)

The extraction analysis script evaluates the binary classification performance of the models. It implements the logic to:

1. Load the ground truth labels (Binary: Contains Claim / No Claim) from the dataset.
2. Parse the model’s JSON output to determine the predicted label (Did the model extract ≥ 1 claim?).
3. Calculate the confusion matrix (TP, FP, TN, FN) across all 150 sentences.
4. Compute the final Precision, Recall, and F1-Scores for both positive and negative classes, aggregating results across the three experimental runs to ensure statistical significance.

4.4.2 Phase 2 Analysis (Verification)

The verification analysis script assesses the accuracy of the models’ fact-checking verdicts. Unlike the binary extraction phase, this is evaluated as a multi-class classification problem. The script:

1. Ingests the benchmark dataset containing the 100 standardized claims and their manual ground truth annotations (*Supported*, *Refuted*, or *Insufficient Info*).
2. Normalizes the model’s text output (e.g., mapping “The claim is false” to the label `Refuted`).
3. Calculates the overall **Accuracy** (percentage of correct verdicts).
4. Computes the **Macro-Averaged F1-Score** to ensure balanced performance across all three truth categories, preventing the metric from being skewed by the majority class.

Chapter 5

Results and Evaluation

This chapter presents the quantitative results of the comparative analysis. The experiments were conducted following the methodology defined in Chapter 3, executing a total of 1,950 API calls (1,350 for extraction across $k=3$ runs, and 600 for verification across $k=2$ runs). All metrics are reported as mean values with standard deviation to demonstrate reliability across runs.

5.1 Phase 1: Claim Extraction Performance

The claim extraction phase evaluated each model’s ability to identify verifiable factual claims from unstructured text. A total of $N = 150$ sentences were processed, with a ground truth distribution of 78 sentences containing factual claims (positive class) and 72 sentences containing non-factual content (negative class).

5.1.1 Quantitative Metric Comparison

Table 5.1 presents the aggregated performance metrics for all three models across the extraction task. The results reveal a distinct precision-recall trade-off pattern that directly addresses **RQ1**.

Model	Accuracy	Precision	Recall	F1-Score
gpt-4o-mini	80.2% \pm 1.7%	83.1% \pm 2.5%	77.8% \pm 0.7%	80.4% \pm 1.5%
gemini-2.5-flash	66.4% \pm 1.5%	89.5% \pm 1.8%	40.2% \pm 2.7%	55.4% \pm 2.8%
deepseek-v3.2	72.7% \pm 2.9%	95.8% \pm 1.7%	49.6% \pm 4.9%	65.3% \pm 4.6%

Table 5.1: Phase 1 Extraction Metrics (Positive Class). Mean \pm standard deviation across $k = 3$ runs. $N = 150$ sentences (78 positive, 72 negative).

Key Finding: gpt-4o-mini achieves the highest F1-Score (80.4%), demonstrating a balanced trade-off between precision and recall. In contrast, gemini-2.5-flash and

deepseek-v3.2 exhibit a conservative extraction strategy, achieving exceptionally high precision (89.5% and 95.8% respectively) at the cost of significantly low recall (40.2% and 49.6%).

This pattern has important practical implications: while gemini-2.5-flash and deepseek-v3.2 rarely misclassify non-factual content as claims (high precision), they fail to identify approximately half of the actual factual claims present in the text (low recall). For a fact-checking pipeline where *coverage* is critical, this represents a significant limitation.

5.1.2 Negative Class Performance

To fully characterize the models’ filtering capabilities, Table 5.2 presents the negative class metrics—measuring each model’s ability to correctly reject non-factual content.

Model	Specificity	NPV	F1-Score (–)
gpt-4o-mini	82.9% \pm 2.9%	77.5% \pm 1.1%	80.1% \pm 1.9%
gemini-2.5-flash	94.9% \pm 0.8%	59.4% \pm 1.1%	73.1% \pm 1.0%
deepseek-v3.2	97.7% \pm 0.8%	64.2% \pm 2.4%	77.5% \pm 2.0%

Table 5.2: Phase 1 Extraction Metrics (Negative Class). Specificity measures the ability to correctly identify non-factual content; NPV (Negative Predictive Value) indicates reliability when predicting “no claim.”

The specificity scores confirm the conservative behavior of gemini-2.5-flash (94.9%) and deepseek-v3.2 (97.7%)—these models almost never incorrectly flag subjective content as factual. However, their lower NPV scores (59.4% and 64.2%) indicate that when they predict “no claim,” there is a substantial probability that they have missed an actual factual claim.

5.1.3 Visual Analysis: Precision-Recall Trade-off

Figure 5.1 visualizes the precision-recall trade-off, with iso-F1 curves providing reference contours. The spatial positioning of each model clearly illustrates the divergent extraction strategies.

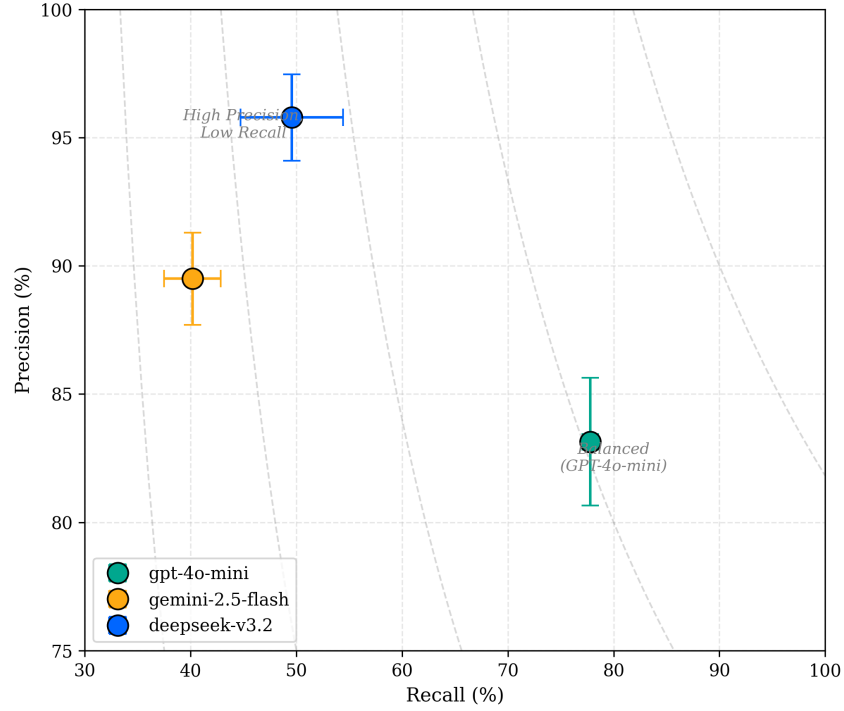


Figure 5.1: Precision-Recall trade-off for claim extraction. gpt-4o-mini occupies the balanced region near $F1=0.8$, while gemini-2.5-flash and deepseek-v3.2 cluster in the high-precision, low-recall quadrant.

5.1.4 Confusion Matrix Analysis

Figure 5.2 presents the averaged confusion matrices across all experimental runs. These matrices provide insight into the specific error patterns of each model.

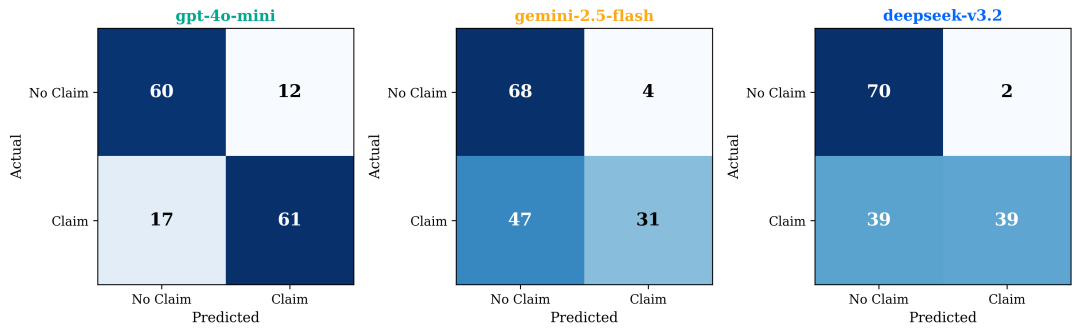


Figure 5.2: Confusion matrices for claim extraction (averaged across $k = 3$ runs). gpt-4o-mini shows balanced errors, while gemini-2.5-flash and deepseek-v3.2 exhibit high false negative counts (47–39 missed claims).

Interpretation:

- **gpt-4o-mini:** Exhibits relatively balanced error distribution with 12 false positives and 17 false negatives on average.

- **gemini-2.5-flash:** Shows extreme conservatism with only 4 false positives but 47 false negatives, missing over half of actual claims.
- **deepseek-v3.2:** Similar conservative pattern with 2 false positives and 39 false negatives.

5.1.5 Reliability Analysis Across Multiple Runs

To assess the stability of each model’s claim extraction performance, Figure 5.3 presents box plots showing the distribution of F1-scores across experimental runs.

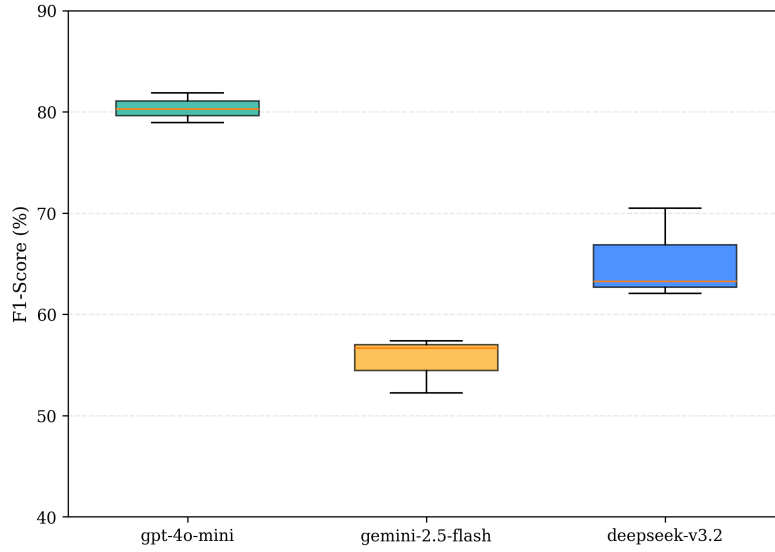


Figure 5.3: Run-to-run variability for extraction phase ($k = 3$ runs). gpt-4o-mini demonstrates the most consistent performance with minimal variance.

gpt-4o-mini demonstrates the most stable performance with the smallest standard deviation ($\pm 1.5\%$ for F1-Score), while deepseek-v3.2 exhibits the highest variability ($\pm 4.6\%$). This variability in deepseek-v3.2’s performance may be attributed to its less deterministic response patterns when processing ambiguous sentences.

5.2 Phase 2: Claim Verification Performance

The verification phase evaluated each model’s ability to assess the veracity of factual claims using search-augmented reasoning. A standardized benchmark of $N = 100$ atomic claims was used, with ground truth labels distributed as: 85 *Supported*, 3 *Refuted*, and 12 *Insufficient Information*.

5.2.1 Quantitative Metric Comparison

Table 5.3 presents the verification performance across all models.

Model	Accuracy	Macro-F1	Supported F1	Refuted F1	Insufficient F1
gpt-4o-mini	79.0% \pm 0.0%	51.2% \pm 3.9%	89.2%	36.7%	27.8%
gemini-2.5-flash	82.0% \pm 1.4%	52.7% \pm 2.0%	90.9%	34.8%	32.5%
deepseek-v3.2	77.5% \pm 2.1%	55.9% \pm 0.4%	87.3%	40.0%	40.5%

Table 5.3: Phase 2 Verification Metrics. Mean \pm standard deviation across $k = 2$ runs. $N = 100$ claims (85 Supported, 3 Refuted, 12 Insufficient).

Key Finding: gemini-2.5-flash achieves the highest accuracy (82.0%), while deepseek-v3.2 achieves the highest Macro-F1 score (55.9%). The divergence between these metrics is explained by the severe class imbalance in the benchmark dataset.

5.2.2 Impact of Class Imbalance

The substantial gap between accuracy and Macro-F1 scores (approximately 25–30 percentage points) reveals a critical limitation: all models perform well on the majority class (*Supported*, $F1 \approx 87\text{--}91\%$) but struggle significantly with minority classes (*Refuted*, $F1 \approx 35\text{--}40\%$; *Insufficient*, $F1 \approx 28\text{--}41\%$).

Figure 5.4 visualizes this class-dependent performance disparity.



Figure 5.4: Per-class F1-scores for verification. The heatmap reveals consistently poor performance on minority classes (Refuted: $n = 3$, Insufficient: $n = 12$) compared to the majority class (Supported: $n = 85$).

Important Caveat: The *Refuted* class contains only 3 samples in the ground truth, making the F1-scores for this class statistically unreliable. This limitation is acknowledged as a threat to validity (see Section 6.3).

5.2.3 Visual Comparison

Figure 5.5 provides a comparative visualization of accuracy versus Macro-F1 for all models.

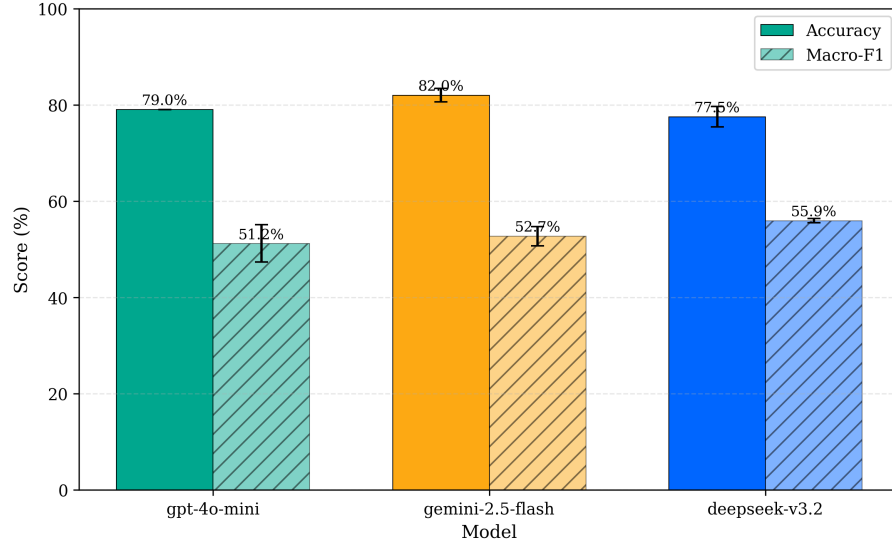


Figure 5.5: Verification performance comparison. The gap between Accuracy and Macro-F1 indicates that high accuracy is driven primarily by majority class performance.

5.2.4 Inter-Model Agreement Analysis

To assess whether models converge on similar verdicts (potentially indicating “easy” vs. “hard” claims), pairwise agreement rates were computed. Figure 5.6 presents these results alongside Fleiss’ Kappa (a statistical measure of inter-rater agreement for categorical data) for overall inter-rater reliability [Landis and Koch, 1977].

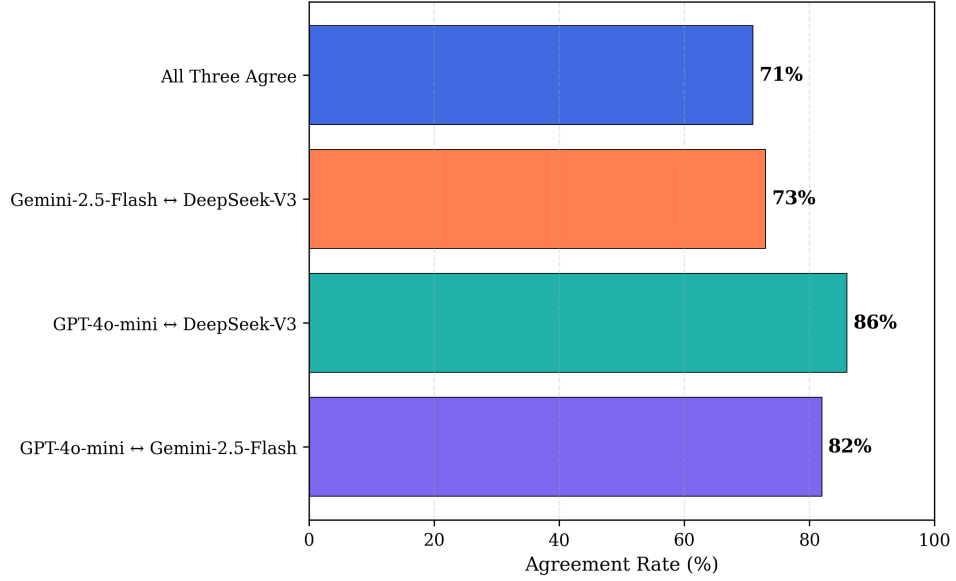


Figure 5.6: Inter-model agreement rates for verification verdicts. Fleiss’ $\kappa = 0.41$ indicates moderate agreement across all three models.

Key Observations:

- gpt-4o-mini and deepseek-v3.2 show the highest pairwise agreement (86%), suggesting similar reasoning patterns.
- gemini-2.5-flash and deepseek-v3.2 show the lowest agreement (73%), indicating divergent decision boundaries.
- All three models agree on 71% of claims, with Fleiss’ $\kappa = 0.41$ (moderate agreement).

The 29% of claims where models disagree represent “ambiguous” cases that warrant qualitative analysis (see Chapter 6).

5.2.5 Reliability Analysis Across Multiple Runs

To assess the stability of each model’s verification performance, Figure 5.7 presents box plots showing the distribution of accuracy scores across experimental runs.

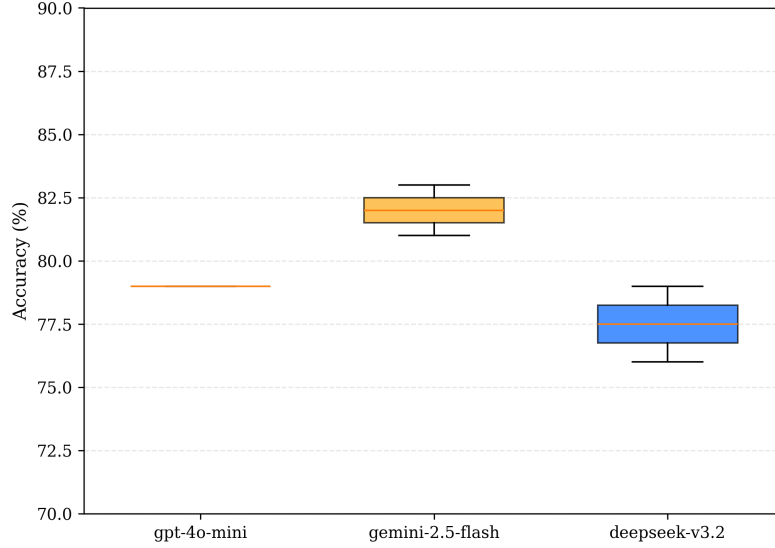


Figure 5.7: Run-to-run variability for verification phase ($k = 2$ runs). gpt-4o-mini produces identical results across runs, while deepseek-v3.2 shows the highest variability.

gpt-4o-mini demonstrates perfect consistency with 0.0% standard deviation in accuracy, producing identical verdicts across independent runs. gemini-2.5-flash shows moderate variability ($\pm 1.4\%$), while deepseek-v3.2 exhibits the highest variability ($\pm 2.1\%$). This pattern mirrors the extraction phase results, confirming that gpt-4o-mini is the most deterministic model across both tasks.

5.3 Summary of Key Findings

Figure 5.8 provides a holistic multi-dimensional comparison of all models across both phases.

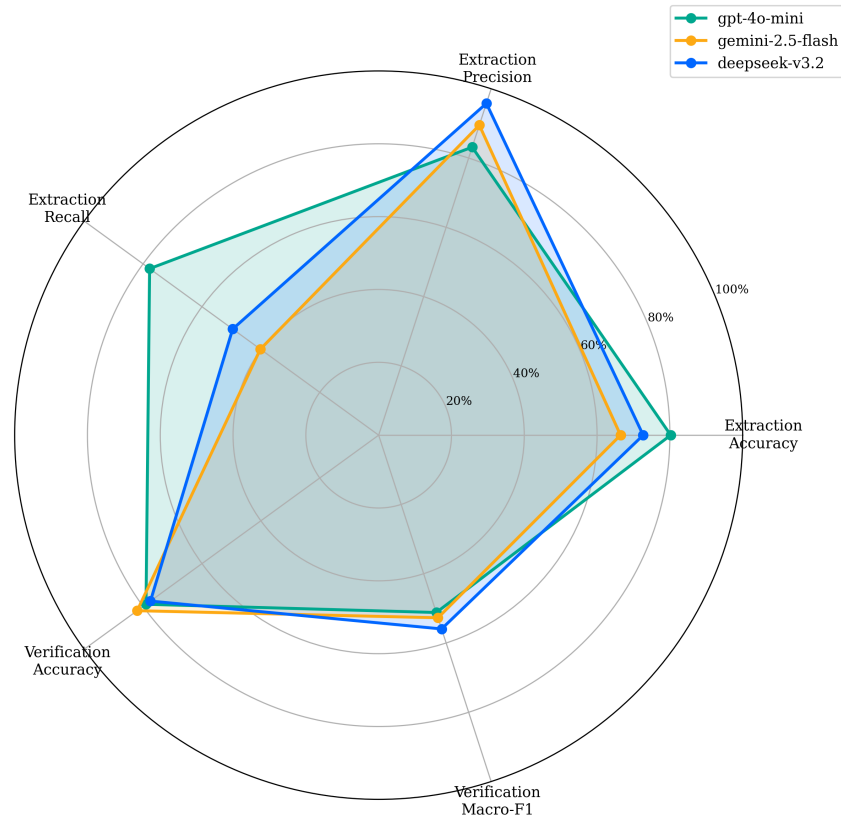


Figure 5.8: Radar chart comparing model performance across all key metrics. gpt-4o-mini demonstrates the most balanced profile, while gemini-2.5-flash and deepseek-v3.2 show specialized strengths.

Table 5.4 synthesizes the key findings from both experimental phases.

Criterion	gpt-4o-mini	gemini-2.5-flash	deepseek-v3.2
Best For	Balanced extraction	High-precision extraction; Verification accuracy	Verification (minority classes)
Extraction F1	80.4% (Best)	55.4%	65.3%
Extraction Strategy	Balanced	Conservative	Conservative
Verification Accuracy	79.0%	82.0% (Best)	77.5%
Verification Macro-F1	51.2%	52.7%	55.9% (Best)
Stability (σ)	Most Stable	Stable	Variable

Table 5.4: Summary of model performance across both experimental phases. Bold indicates best performance for each criterion.

Chapter 6

Discussion

This chapter interprets the empirical findings presented in Chapter 5, contextualizes them within the broader literature, and provides a qualitative analysis of model failure modes. We adopt a useful analogy throughout: if these LLMs were members of a legal research team, **gpt-4o-mini** would be the balanced researcher who finds most evidence, while **gemini-2.5-flash** and **deepseek-v3.2** are the overly cautious lawyers who only cite a fact if they are highly confident—even if it means ignoring half of the relevant material.

6.1 Comparative Analysis of LLMs

6.1.1 Best Performing LLM for Extraction

The extraction phase results clearly establish **gpt-4o-mini** as the most effective model for claim extraction, achieving an F1-Score of 80.4% compared to 55.4% (**gemini-2.5-flash**) and 65.3% (**deepseek-v3.2**). This superiority stems from **gpt-4o-mini**’s balanced approach to the precision-recall trade-off.

The critical distinction lies in *extraction strategy*:

- **gpt-4o-mini (Balanced)**: Achieves 83.1% precision and 77.8% recall, correctly identifying approximately 8 out of 10 actual factual claims while maintaining high precision.
- **gemini-2.5-flash & deepseek-v3.2 (Conservative)**: Achieve exceptionally high precision (89.5% and 95.8% respectively) but at severe cost to recall (40.2% and 49.6%). These models fail to identify approximately half of all factual claims present in the text.

This conservative behavior has significant practical implications. In a fact-checking

pipeline, a missed claim (false negative) represents information that will never be verified—a silent failure that undermines the system’s utility. The high specificity of gemini-2.5-flash (94.9%) and deepseek-v3.2 (97.7%) indicates excellent ability to filter out subjective content, but their low Negative Predictive Value (NPV: 59.4% and 64.2%) reveals that when these models classify a sentence as “non-factual,” there is approximately a 35–40% chance it actually contained a verifiable claim.

Recommendation: For extraction tasks where comprehensive coverage is important, gpt-4o-mini is the clear choice. gemini-2.5-flash and deepseek-v3.2 may be appropriate only when precision is paramount and missing claims is acceptable.

6.1.2 Best Performing LLM for Verification

The verification phase presents a more nuanced picture, with different models excelling on different metrics:

- **gemini-2.5-flash** achieves the highest accuracy (82.0%), making it the best choice when the primary concern is overall correctness on a representative dataset.
- **deepseek-v3.2** achieves the highest Macro-F1 (55.9%), indicating superior performance on minority classes despite lower overall accuracy.
- **gpt-4o-mini** demonstrates the highest determinism (0.0% standard deviation in accuracy), producing identical results across runs.

The 25–30 percentage point gap between accuracy and Macro-F1 scores across all models highlights the statistical impact of class imbalance. While models perform well on the majority class (*Supported*), their performance metrics on minority classes (*Refuted*, *Insufficient Information*) are volatile due to the small sample size ($n=3$ for *Refuted*), making it difficult to conclusively determine whether the lower scores reflect a lack of reasoning capability or a statistical artifact.

Comparison with SAFE: Wei et al. [Wei et al., 2024] reported that their SAFE method achieves 72% agreement with human annotators while often surpassing humans in precision through more targeted search queries. Our results show comparable patterns: all three models achieve 77–82% accuracy on claims where human annotators established ground truth, suggesting that LLM-based verification has reached a level of reliability suitable for practical deployment, at least for majority-class claims.

6.1.3 The Stability-Performance Trade-off

An unexpected finding was the relationship between model stability and performance variability:

- **gpt-4o-mini:** Most stable ($\pm 0.0\%$ accuracy SD, $\pm 1.5\%$ extraction F1 SD)
- **gemini-2.5-flash:** Moderately stable ($\pm 1.4\%$ accuracy SD, $\pm 2.8\%$ extraction F1 SD)
- **deepseek-v3.2:** Most variable ($\pm 2.1\%$ accuracy SD, $\pm 4.6\%$ extraction F1 SD)

deepseek-v3.2’s higher variability may be attributed to its sensitivity to prompt formatting. As noted in Chapter 4, deepseek-v3.2 required custom middleware to enforce structured JSON output, suggesting that Open Weights models may require additional engineering effort to achieve production-level consistency.

6.2 Qualitative Error Analysis

To understand *why* models fail, we analyzed the 29% of verification cases (29 out of 100 claims) where at least two models produced different verdicts.

6.2.1 Common Failure Patterns in Claim Extraction

Analysis of extraction errors revealed three primary failure modes:

- 1. Over-Filtering of Context-Dependent Claims:** gemini-2.5-flash and deepseek-v3.2 frequently rejected claims that required contextual interpretation. For example, sentences containing demonstrative pronouns (“This process...”, “These results...”) were often classified as non-factual despite containing verifiable information.
- 2. Ambiguous Factuality Boundaries:** All models struggled with claims at the boundary between factual and subjective content. Statements like “Improving sleep quality is a common goal for many people” were inconsistently classified—gpt-4o-mini tended to accept such claims as factual (high recall), while gemini-2.5-flash and deepseek-v3.2 rejected them (high precision).
- 3. Markdown Formatting Artifacts:** The BingCheck dataset contains Markdown formatting (e.g., “**bold text**”, bullet points). Some models were confused by these artifacts, occasionally treating formatting as content or failing to parse claims embedded in list structures.

6.2.2 Common Failure Patterns in Claim Verification

Analysis of the **reasoning** field in model outputs revealed distinct failure patterns:

- 1. Over-Cautious “Insufficient Information” Verdicts:** The most common error pattern was models returning *Insufficient Information* for claims that were actually *Supported*. This occurred in:

- gpt-4o-mini: 10 cases (10% of claims)
- gemini-2.5-flash: 2 cases (2% of claims)
- deepseek-v3.2: 18 cases (18% of claims)

Interestingly, this pattern is inverse to extraction behavior: deepseek-v3.2, which was the most conservative extractor (highest precision), is also the most cautious verifier. This suggests a consistent “personality” characterized by epistemic caution.

2. Search Query Quality: Examination of the reasoning traces revealed that verification failures often originated in the search phase rather than the reasoning phase. When models generated overly specific or poorly-phrased search queries, they retrieved irrelevant evidence and subsequently issued incorrect verdicts. For example, for the claim “The hot climate caused around 15 million gallons of water to condense from the structure,” all models struggled to find relevant sources because the claim lacked sufficient context about *which* structure was being referenced.

3. Temporal and Specificity Challenges: Claims containing specific dates, quantities, or proper nouns proved difficult when web search results did not contain exact matches. The claim “The Mayans invented the concept independently circa 4 A.D.” (ground truth: *Refuted*) produced three different verdicts: gpt-4o-mini returned *Insufficient Information*, gemini-2.5-flash correctly identified it as *Refuted*, and deepseek-v3.2 incorrectly classified it as *Supported*.

6.2.3 The 69% Unanimous Correctness

While all three models *agreed* on 71% of claims (Figure 5.6), they were unanimously *correct* on 69%—meaning 2 claims saw all models confidently agree on an incorrect verdict. This distinction between agreement and correctness is important: high inter-model agreement does not guarantee accuracy.

Nevertheless, the 69% unanimous correctness rate is a positive finding. It suggests that for straightforward factual claims with clear web evidence, current LLMs have achieved reliable verification capability. The remaining 31% of claims represent genuinely difficult cases—either due to ambiguous evidence, context-dependent claims, or limitations in web search coverage.

6.3 Limitations of the Study

6.3.1 Sample Size and Domain Specificity

While $N = 150$ sentences for extraction and $N = 100$ claims for verification provide sufficient statistical power for comparative analysis, several limitations must be acknowledged:

1. Class Imbalance: The verification benchmark contains 85 *Supported*, 12 *Insufficient Information*, and only 3 *Refuted* claims. This severe imbalance means that:

- Accuracy metrics are dominated by majority-class performance
- Per-class F1-scores for *Refuted* (based on $n = 3$) are statistically unreliable and should be interpreted as indicative rather than definitive
- The models’ true ability to detect misinformation cannot be conclusively assessed from this dataset

2. Domain Generalization: The BingCheck dataset represents general-knowledge claims from Bing Chat responses. Results may not generalize to specialized domains such as:

- Medical claims (requiring clinical evidence)
- Legal statements (requiring jurisdictional specificity)
- Scientific claims (requiring peer-reviewed source evaluation)

3. Temporal Validity: Web search results change over time. A claim that was verifiable during our experiments (December 2025) may produce different search results in the future, potentially affecting reproducibility.

6.3.2 Constraints of the TruthLens Architecture

1. Single Search Provider: All experiments used Brave Search exclusively. Different search APIs (Google, Bing, Exa) may retrieve different evidence, potentially affecting verification outcomes.

2. Fixed Iteration Limits: The verification agent uses a maximum of 5 search iterations. Some claims may require more extensive evidence gathering than this limit permits.

3. Binary Extraction Ground Truth: The extraction phase uses binary labels (factual/non-factual) from BingCheck. This simplification does not capture the

spectrum of factuality (e.g., partially factual claims, opinions presented as facts).

4. Model Version Sensitivity: LLM capabilities evolve rapidly. The specific model versions tested (gpt-4o-mini, Gemini-2.5-flash, deepseek-v3.2) represent a snapshot in time; newer versions may exhibit different performance characteristics.

6.3.3 Threats to Validity

Internal Validity: The use of multiple independent runs ($k = 3$ for extraction, $k = 2$ for verification) mitigates concerns about result variability, but the limited number of runs for verification (due to API costs) means that standard deviation estimates for that phase are less precise.

External Validity: Results are specific to the English language, the BingCheck dataset’s claim distribution, and the particular prompt templates used in the TruthLens agents. Generalization to other languages, domains, or prompt formulations requires further investigation.

Construct Validity: The metrics used (F1-Score, Accuracy, Macro-F1) are standard in NLP evaluation but may not fully capture the practical utility of a fact-checking system. Real-world deployment would require additional metrics such as user trust, explanation quality, and end-to-end latency.

Chapter 7

Conclusion and Future Work

This thesis presented a comparative analysis of three Large Language Models—gpt-4o-mini, gemini-2.5-flash, and deepseek-v3.2—within the TruthLens framework for automated fact-checking. Through rigorous experimentation across 1,950 API calls, we evaluated model performance on both claim extraction and verification tasks. This chapter summarizes the key findings, articulates the contributions of this work, offers practical guidance for deployment, and outlines directions for future research.

7.1 Summary of Findings

The experimental results provide clear answers to the three research questions posed in Chapter 1.

7.1.1 Answering Research Question 1 (Extraction)

RQ1: *Which LLM most accurately identifies verifiable factual claims from text at the sentence level?*

Answer: **gpt-4o-mini** emerges as the most effective model for claim extraction, achieving an F1-Score of **80.4%** compared to 55.4% (gemini-2.5-flash) and 65.3% (deepseek-v3.2).

The critical distinction lies in extraction strategy. gpt-4o-mini adopts a *balanced approach*, achieving 83.1% precision and 77.8% recall—correctly identifying approximately 8 out of 10 factual claims while maintaining high precision. In contrast, gemini-2.5-flash and deepseek-v3.2 employ a *conservative strategy*, achieving exceptionally high precision (89.5% and 95.8% respectively) but at severe cost to recall (40.2% and 49.6%). These models fail to identify approximately half of all factual claims present in the input text.

For fact-checking pipelines where *comprehensive coverage* is a high priority, gpt-4o-mini is the clear choice. The conservative models may be appropriate only in specialized contexts where false positives carry higher costs than missed claims.

7.1.2 Answering Research Question 2 (Verification)

RQ2: *Which LLM most accurately assesses the veracity of identified claims when provided with claims?*

Answer: The verification phase presents a nuanced picture depending on the evaluation metric:

- **gemini-2.5-flash** achieves the highest accuracy (**82.0%**), making it the optimal choice when overall correctness on representative datasets is the primary concern.
- **deepseek-v3.2** achieves the highest Macro-F1 (**55.9%**), indicating superior balanced performance across all verdict categories, including minority classes.
- **gpt-4o-mini** demonstrates the highest determinism (0.0% standard deviation), producing identical results across independent runs.

The substantial gap between accuracy (77–82%) and Macro-F1 (50–56%) across all models highlights the impact of dataset imbalance. While models demonstrate high reliability on the majority class (*Supported*, $F1 \approx 87\text{--}91\%$), the shortage of minority samples ($n=3$ for *Refuted*) results in unpredictable metrics where single errors disproportionately penalize performance. This discrepancy suggests that while models show bias toward the majority class, their true capability in detecting misinformation requires further evaluation on more balanced datasets.

7.1.3 Answering Research Question 3 (Qualitative)

RQ3: *What are the common failure modes in LLM-based fact-checking?*

The qualitative error analysis identified three primary failure patterns:

- 1. Over-Filtering in Extraction:** gemini-2.5-flash and deepseek-v3.2 frequently rejected context-dependent claims containing demonstrative pronouns (“This process...”, “These results...”), despite containing verifiable information. This conservative behavior contributed to their low recall scores.
- 2. Search Query Quality:** Verification failures often originated in the search phase rather than the reasoning phase. Poorly-phrased or overly-specific search queries led to irrelevant evidence retrieval, subsequently causing incorrect verdicts. Claims

lacking sufficient context (e.g., references to “the structure” without specifying which structure) were particularly problematic.

3. Class Imbalance Sensitivity: All models exhibited bias toward the majority class (*Supported*), frequently returning over-cautious *Insufficient Information* verdicts for claims that were actually supported. deepseek-v3.2 showed this pattern most prominently (18% of claims), while gemini-2.5-flash was least affected (2%).

Despite these challenges, the models achieved 69% unanimous correctness across all verification cases, demonstrating that for straightforward factual claims with clear web evidence, LLM-based verification has reached a level of reliability suitable for practical deployment.

7.2 Contributions

This thesis makes three primary contributions to the field of automated fact-checking:

1. Infrastructure for Systematic Multi-Model Benchmarking: We extended the open-source ClaimeAI framework with systematic infrastructure for controlled comparative analysis. The engineering contributions include:

- A unified provider management layer ensuring experimental consistency across models with different API capabilities
- Custom structured output middleware for DeepSeek models, bridging the gap between basic JSON mode and Pydantic schema validation
- Automated batch processing pipeline supporting large-scale execution ($N = 150$ samples, $k = 3$ runs, 3 models) with rate limiting and failure recovery
- Integration of cost-effective Brave Search API for scalable evidence retrieval
- Multi-run statistical validation framework for quantifying performance variability

2. Empirical Comparative Analysis: We conducted a systematic comparison of gpt-4o-mini, gemini-2.5-flash, and deepseek-v3.2 within a decoupled two-phase fact-checking architecture. The experimental design—with $k = 3$ runs for extraction and $k = 2$ runs for verification across 1,950 total API calls—provides performance estimates with quantified variability. The results establish clear performance hierarchies: gpt-4o-mini for extraction (balanced precision-recall), gemini-2.5-flash for verification accuracy, and deepseek-v3.2 for balanced multi-class verification.

3. Methodological Contribution: We demonstrated the value of decoupled evaluation in diagnosing LLM capabilities. By separating extraction from verification,

we identified that the same model can exhibit different behavioral patterns across tasks (e.g., deepseek-v3.2’s conservative extraction correlates with cautious verification). This granular analysis would be less feasible with monolithic end-to-end evaluation approaches.

7.3 Practical Implications

Based on the empirical findings, we offer the following guidance for practitioners deploying LLM-based fact-checking systems:

Model Selection by Use Case:

- **High-Coverage Fact-Checking:** Use gpt-4o-mini for extraction to maximize claim identification (77.8% recall), accepting moderate false positives.
- **High-Precision Applications:** Use gemini-2.5-flash or deepseek-v3.2 for extraction when false positives are costly, understanding that approximately half of factual claims will be missed.
- **Verification Tasks:** Use gemini-2.5-flash for highest overall accuracy (82.0%) on representative datasets, or deepseek-v3.2 when balanced performance across verdict categories is required.

Stability Considerations: gpt-4o-mini offers the most deterministic outputs ($\pm 0.0\%$ accuracy SD), making it suitable for applications requiring reproducible results. deepseek-v3.2’s higher variability ($\pm 4.6\%$ extraction F1 SD) suggests it may require additional engineering effort (e.g., ensemble approaches or multiple inference passes) to achieve production-level consistency.

Cost-Accuracy Trade-offs: Open Weights models like deepseek-v3.2 offer competitive performance at reduced API costs compared to proprietary alternatives. For budget-constrained deployments, deepseek-v3.2 provides a viable option, particularly for verification tasks where its Macro-F1 performance exceeds the other models.

7.4 Future Work

The findings of this thesis suggest several promising directions for future research:

7.4.1 Alternative Search Providers

This study exclusively utilized the Brave Search API due to budget constraints. Future work should investigate the impact of different search providers on verification accuracy:

- **Neural Search APIs** (Exa, Tavily) may retrieve more semantically relevant evidence compared to keyword-based approaches.
- **Major Search Engines** (Google, Bing) offer broader web coverage but at higher cost and with different result ranking algorithms.

A systematic comparison across search providers would isolate the contribution of evidence retrieval quality to overall verification performance.

7.4.2 Multi-Agent Consensus Architectures

A significant limitation of the current architecture is single-agent decision-making. Future work should explore **multi-agent consensus mechanisms**:

- **Judge Agents:** Introducing a separate LLM to review and validate the decisions of the extraction and verification agents before finalizing outputs.
- **Unanimous Decision Requirements:** Requiring agreement across multiple independent agents before accepting a verdict, similar to ensemble methods in machine learning.
- **Adversarial Review:** Implementing agents that specifically attempt to refute or challenge preliminary verdicts, forcing more rigorous evidence evaluation.

While these approaches increase computational cost and API expenses, they may substantially improve accuracy and reliability—particularly for the minority classes where current models struggle.

7.4.3 Self-Correction Mechanisms

A promising direction for improving verification reliability is the integration of **self-correction mechanisms**, where models iteratively refine their outputs based on feedback signals. Recent research has demonstrated that LLMs can improve their reasoning through structured self-reflection loops, most notably the Chain-of-Verification (CoVe) method [Dhuliawala et al., 2024], which prompts models to draft initial responses, generate verification questions, and independently answer those questions to identify inconsistencies.

Potential implementations include:

- **Confidence-Based Re-evaluation:** When a model’s confidence score falls below a threshold, automatically triggering additional search iterations or alternative query formulations. Varshney et al. [Varshney et al., 2023] demonstrated that validating low-confidence generations can effectively detect and mitigate hallucinations before they propagate.

- **Contradiction Detection:** Implementing logic to identify when retrieved evidence contradicts the model’s preliminary verdict, forcing explicit reconsideration—a core principle of the CoVe framework [Dhuliawala et al., 2024].
- **Evidence Quality Assessment:** Training models to evaluate the reliability and relevance of retrieved sources before incorporating them into reasoning. Pan et al. [Pan et al., 2024] introduced Credibility-aware Generation (CAG), teaching LLMs to weight evidence based on source credibility.

Such mechanisms could particularly benefit minority class detection, where the current models show the weakest performance. By requiring explicit justification before finalizing verdicts on potentially *Refuted* or *Insufficient Information* claims, self-correction could reduce the over-cautious behavior observed in deepseek-v3.2 and improve the overall Macro-F1 scores.

7.4.4 Adversarial Robustness

A critical dimension not explored in this thesis is the vulnerability of automated fact-checking systems to adversarial attacks. As Liu et al. [Liu et al., 2025] comprehensively survey, fact-checking pipelines face threats from adversaries who manipulate claims, evidence, or claim-evidence pairs to distort verification outcomes. Such attacks can exploit weaknesses in both the extraction and verification phases, potentially causing systems to misclassify false claims as supported or vice versa.

Future research should investigate:

- **Attack Surface Analysis:** Systematic evaluation of how TruthLens-style architectures respond to adversarially crafted inputs, including paraphrased claims designed to evade detection and poisoned evidence injected into search results.
- **Robustness Testing:** Benchmarking model performance under adversarial conditions, measuring the degradation in accuracy when claims are subtly modified while preserving their semantic meaning.
- **Defense Mechanisms:** Developing adversary-aware protection strategies, such as ensemble verification across multiple models or anomaly detection for suspicious claim patterns.

As automated fact-checking systems are deployed in high-stakes environments, understanding and mitigating adversarial vulnerabilities becomes essential for maintaining user trust and system reliability.

7.4.5 Domain-Specific and Multilingual Extensions

This study evaluated models on the BingCheck dataset, which represents general-knowledge claims across diverse but primarily English-language domains. A significant limitation is the lack of evaluation on specialized domain-specific and multilingual benchmarks. Recent research has produced several datasets that could extend the generalizability of findings:

Domain-Specific Datasets:

- **Scientific Claims:** The SciFact-Open dataset [Wadden et al., 2022] provides a large-scale benchmark of scientific claims against 500,000 research abstracts, revealing that systems trained on smaller corpora experience performance drops of at least 15 F1 points when applied to realistic retrieval settings.
- **Health Misinformation:** COVID-Fact [Saakyan et al., 2021] offers 4,086 pandemic-related claims with evidence, addressing the critical domain of health misinformation where verification errors carry significant real-world consequences.
- **Climate Claims:** CLIMATE-FEVER [Diggelmann et al., 2020] adapts the FEVER methodology—the widely-adopted benchmark framework that pairs claims with Wikipedia evidence sentences and three-way veracity labels (Supported, Refuted, Not Enough Info) [Thorne et al., 2018]—to 1,535 real-world climate statements, demonstrating the “subtle complexity” of modeling authentic climate discourse.

Multilingual Benchmarks:

- **Cross-Lingual Evaluation:** X-Fact [Gupta and Srikumar, 2021] provides expert-labeled claims across 25 languages, enabling evaluation of zero-shot multilingual capabilities—where models are tested on languages not seen during training, assessing their ability to transfer fact-checking skills across linguistic boundaries.
- **Non-English Languages:** CHEF [Hu et al., 2022] offers 10,000 Chinese claims with annotated evidence, addressing the gap in non-English fact-checking resources and demonstrating that “misinformation spans both geographic and linguistic boundaries.”

Future work should evaluate the TruthLens architecture on these specialized benchmarks to assess whether the performance patterns observed in this thesis—particularly the precision-recall trade-offs and minority class challenges—generalize across domains and languages.

7.4.6 Real-World Deployment and User Studies

The ultimate test of any fact-checking system is its utility to real users. Future work should include:

- **Browser Extension Deployment:** The TruthLens framework already includes a prototype browser extension. Deploying this to real users would provide invaluable feedback on practical usability, latency requirements, and explanation quality.
- **User Trust Studies:** Investigating how users perceive and interact with LLM-generated fact-checking verdicts, including how explanation quality affects user trust.
- **Iterative UX Refinement:** Using real-world feedback to guide interface improvements, determining how to present verdicts, confidence scores, and evidence sources in ways that users find helpful and actionable.

Such studies would inform the productization of LLM-based fact-checking tools and identify the gap between technical performance metrics and genuine user benefit.

7.5 Closing Remarks

The rise of Large Language Models has created both the problem and the potential solution for automated fact-checking. These models generate fluent but sometimes factually incorrect content at unprecedented scale, yet they also possess the reasoning capabilities to verify claims against external evidence.

This thesis demonstrates that current LLMs, when deployed within a structured agentic framework, can achieve human-comparable performance on straightforward factual claims. gpt-4o-mini’s 80.4% extraction F1 and gemini-2.5-flash’s 82.0% verification accuracy represent meaningful progress toward reliable automated fact-checking.

However, significant challenges remain. The struggle with minority classes, the sensitivity to search query quality, and the 31% of claims where models disagree all indicate that LLM-based fact-checking is not yet ready to operate without human oversight. Crucially, while models achieved 69% unanimous correctness, the remaining 31% of ambiguous cases—where even state-of-the-art models produce conflicting verdicts—underscore that **human-in-the-loop validation remains essential** for high-stakes fact-checking applications. The path forward lies in hybrid systems that leverage LLM capabilities while acknowledging their limitations—using multi-agent architectures, diverse evidence sources, and human review for contested or

high-consequence decisions.

As misinformation continues to proliferate in the digital information ecosystem, the development of trustworthy, transparent, and accessible fact-checking tools becomes increasingly critical. This thesis contributes a step toward that goal, providing both empirical evidence of current capabilities and a foundation for future improvements.

Appendix A

Appendix

A.1 Full Benchmark Results Tables

This section presents the complete experimental results across all independent runs, providing full transparency for reproducibility.

A.1.1 Phase 1: Extraction Results Per Run

Table A.1 presents the detailed extraction metrics for each of the $k = 3$ experimental runs.

Table A.1: Claim Extraction Metrics Per Run ($N = 150$ sentences)							
Model	Run	Accuracy	Precision	Recall	F1-Score	TP	FN
gpt-4o-mini	Run 1	78.7%	81.1%	76.9%	78.9%	60	18
	Run 2	82.0%	85.9%	78.2%	81.9%	61	17
	Run 3	80.0%	82.4%	78.2%	80.3%	61	17
gemini-2.5-flash	Run 1	64.7%	87.9%	37.2%	52.3%	29	49
	Run 2	67.3%	89.2%	42.3%	57.4%	33	45
	Run 3	67.3%	91.4%	41.0%	56.6%	32	46
deepseek-v3.2	Run 1	76.0%	97.7%	55.1%	70.5%	43	35
	Run 2	70.7%	94.7%	46.2%	62.1%	36	42
	Run 3	71.3%	94.9%	47.4%	63.2%	37	41

A.1.2 Phase 2: Verification Results Per Run

Table A.2 presents the detailed verification metrics for each of the $k = 2$ experimental runs.

A.1.3 Confusion Matrices

Table A.3 presents the averaged confusion matrices for the extraction phase.

Table A.2: Claim Verification Metrics Per Run ($N = 100$ claims)

Model	Run	Accuracy	Macro-F1	Supported F1	Insufficient F1
gpt-4o-mini	Run 1	79.0%	48.5%	89.8%	22.2%
	Run 2	79.0%	53.9%	88.5%	33.3%
gemini-2.5-flash	Run 1	83.0%	54.1%	90.7%	35.3%
	Run 2	81.0%	51.3%	91.0%	29.6%
deepseek-v3.2	Run 1	76.0%	55.6%	85.9%	41.0%
	Run 2	79.0%	56.3%	88.8%	40.0%

Table A.3: Averaged Confusion Matrices for Claim Extraction ($k = 3$ runs)

	gpt-4o-mini		gemini-2.5-flash		deepseek-v3.2	
	Pred +	Pred −	Pred +	Pred −	Pred +	Pred −
Actual +	60.7	17.3	31.3	46.7	38.7	39.3
Actual −	12.3	59.7	3.7	68.3	1.7	70.3

A.1.4 Inter-Model Agreement Analysis

Table A.4 presents the pairwise agreement rates and error analysis for the verification phase.

Table A.4: Inter-Model Agreement and Error Analysis (Verification Phase)

Metric	Value
gpt-4o-mini \leftrightarrow gemini-2.5-flash Agreement	82%
gpt-4o-mini \leftrightarrow deepseek-v3.2 Agreement	86%
gemini-2.5-flash \leftrightarrow deepseek-v3.2 Agreement	73%
All Three Models Agree	71%
Fleiss’ Kappa (κ)	0.41
Error Patterns (Run 1)	
gpt-4o-mini Over-Cautious Errors	10
gemini-2.5-flash Over-Cautious Errors	2
deepseek-v3.2 Over-Cautious Errors	18

A.2 Prompt Templates

This section documents excerpts from the system prompts used in the TruthLens framework. These prompts are **not original contributions** of this thesis; they are adapted from the open-source *ClaimAI* repository [BharathxD, 2024], which implemented methodologies from two foundational papers:

- **Extraction Prompts:** Based on the *Claimify* framework by Metropolitansky & Larson [Metropolitansky and Larson, 2025], which established the selection-disambiguation-decomposition pipeline for atomic claim extraction.
- **Verification Prompts:** Based on the *Search-Augmented Factuality Evaluator (SAFE)* by Wei et al. [Wei et al., 2024], which introduced the iterative search-reason-verify paradigm.

The complete prompts with all examples are available in the project repository (apps/agent/claim_extractor/prompts.py and apps/agent/claim_verifier/prompts.py). Below are condensed excerpts showing the core instructions.

A.2.1 Claim Extraction Prompts

Selection Prompt

The selection prompt determines whether a sentence contains verifiable factual claims:

Selection System Prompt (Excerpt)

You are an assistant to a fact-checker. You will be given an excerpt from a text and a particular sentence of interest. Your task is to determine whether this particular sentence contains at least one specific and verifiable proposition.

Note the following rules:

- If the sentence is about a lack of information, it does NOT contain a specific and verifiable proposition.
- It does NOT matter whether the proposition is true or false.
- Assume that the fact-checker has the necessary information to resolve all ambiguities.

Here are some examples of sentences that do NOT contain any specific and verifiable propositions:

- By prioritizing ethical considerations, companies can ensure that their innovations are socially responsible
- Technological progress should be inclusive
- AI could lead to advancements in healthcare

Here are some examples of sentences that likely contain a specific and verifiable proposition:

- The partnership between Company X and Company Y illustrates the power of innovation -> "There is a partnership between Company X and Company Y"
- Jane Doe's approach includes embracing adaptability and prioritizing customer feedback

Disambiguation Prompt

The disambiguation prompt resolves referential and structural ambiguities:

Disambiguation System Prompt (Excerpt)

You are an assistant to a fact-checker. Your task is to “decontextualize” the sentence, which means:

1. Resolve partial names and undefined acronyms/abbreviations
2. Resolve linguistic ambiguity that has a clear resolution

“Linguistic ambiguity” refers to the presence of multiple possible meanings in a sentence. Vagueness and generality are NOT linguistic ambiguity.

Example:

- Context: “John Smith was an early employee who transitioned to management in 2010”
- Sentence: “At the time, he led the company’s operations and finance teams.”
- Decontextualized: “In 2010, John Smith led the company’s operations team and finance team.”

Decomposition Prompt

The decomposition prompt breaks complex sentences into atomic claims:

Decomposition System Prompt (Excerpt)

You are an assistant for a group of fact-checkers. Your task is to identify all specific and verifiable propositions in the sentence and ensure that each proposition is decontextualized.

A proposition is “decontextualized” if:

1. It is fully self-contained (can be understood in isolation)
2. Its meaning in isolation matches its meaning when interpreted alongside the context

The propositions should be the simplest possible discrete units of information.

Example:

- Sentence: “John Smith and Jane Doe (writers of ‘Fighting for Better Tech’)”
- Propositions: [“John Smith is a writer of ‘Fighting for Better Tech’”, “Jane Doe is a writer of ‘Fighting for Better Tech’”]

A.2.2 Claim Verification Prompts

Query Generation Prompt

Query Generation System Prompt (Excerpt)

You are an expert search query generator for fact-checking claims.

Your task: Create a single, effective search query to find evidence that could verify or refute the given claim.

Requirements:

- Include key entities, names, dates, and specific details
- Use search-engine-friendly language (no special characters)
- Target authoritative sources (news, government, academic)
- Keep it concise (5-15 words optimal)
- Design to find both supporting AND contradictory evidence

Examples:

- Policy claim: “Biden student loan forgiveness program 2023”
- Statistics: “unemployment rate March 2024 Bureau Labor”
- Events: “Taylor Swift concert cancellation official statement”

Evidence Evaluation Prompt

Evidence Evaluation System Prompt (Excerpt)

You are an expert fact-checker. Evaluate claims based ONLY on the evidence provided--do not use prior knowledge.

Verdict criteria:

SUPPORTED--Use when:

- Multiple reliable sources confirm the claim
- Evidence directly addresses the core assertion
- No credible contradictory evidence

REFUTED--Use when:

- Authoritative sources explicitly contradict the claim
- Evidence provides clear counter-factual information

INSUFFICIENT INFORMATION--Use when:

- Limited evidence (too few sources)
- Evidence is indirect, vague, or incomplete

- Key information missing for verification

Decision rule: Be conservative--when evidence is ambiguous, choose “Insufficient Information.”

A.3 Model Configuration

To ensure reproducibility, exact model versions and generation parameters were standardized across all experimental runs.

A.3.1 Provider Settings

Table A.5 documents the exact model identifiers and API endpoints used.

Table A.5: LLM API Configuration (exact model identifiers used in code)

Display Name	API Model ID (in code)	Provider Endpoint
gpt-4o-mini	openai:gpt-4o-mini	OpenAI API (api.openai.com)
gemini-2.5-flash	gemini-2.5-flash	Google AI API (generativelanguage.googleapis.com)
deepseek-v3.2	deepseek-chat	DeepSeek API (https://api.deepseek.com)

A.3.2 Generation Parameters

To minimize non-determinism, the generation parameters in Table A.6 were enforced at the code level (via the `LLMProvider` class).

Table A.6: LLM Generation and network parameters (as implemented in code)

Parameter	Value (code)
Temperature (default for deterministic runs)	0.0
Network retry attempts (search client)	2 retries (up to 3 attempts total)
Request timeout (Brave search requests)	30 seconds (per-request aio-http timeout)
JSON / structured-output handling	Enabled for DeepSeek via <code>DeepSeekChatWrapper</code> ; OpenAI/Gemini use provider-native mechanisms or client support

A.3.3 Search Configuration

- **Search Provider:** Brave Search API (default). The implementation supports alternative providers (Exa, Tavily) via the `SEARCH_PROVIDER` environment variable.
- **Maximum Search Iterations:** 5 per claim (default configured in `claim_verifier/config/nodes.py`)
- **Results per Query:** 3 (default configured in `claim_verifier/config/nodes.py`; configurable)
- **Freshness Filter:** None (all time) by default

A.3.4 Environment Variables

The following environment variables must be configured to run the benchmark:

```
LLM_PROVIDER=openai|gemini|deepseek
OPENAI_API_KEY=sk-proj-...
GOOGLE_API_KEY=AIza...
DEEPSEEK_API_KEY=sk-...
BRAVE_API_KEY=your-brave-key-here
EXA_API_KEY=<uuid4-or-EXA-key>
TAVILY_API_KEY=tvly-...
SEARCH_PROVIDER=brave|exa|tavily
```

A.4 Benchmark Runner Script

The following Python script orchestrates the experimental pipeline. The complete implementation is available in the project repository.

```
#!/usr/bin/env python3
"""
Script to run the extraction phase for all three LLMs
on the thesis dataset with per-sentence updates and
resume capability for cost protection.
"""

import asyncio
import pandas as pd
from pathlib import Path
```

```

# Provider mapping
PROVIDERS = {
    'gpt4': 'openai',
    'gemini': 'gemini',
    'deepseek': 'deepseek'
}

async def run_extraction_for_sentence(sentence: str,
                                     provider: str) -> dict:
    """
    Run claim extraction for a single sentence using
    the specified LLM provider.
    """
    from utils.settings import settings
    settings.llm_provider = provider

    payload = {
        "answer_text": sentence,
        "metadata": f"extraction-{provider}"
    }

    result = await claim_extractor_graph.ainvoke(payload)
    validated_claims = result.get('validated_claims', [])

    return {
        'binary_result': len(validated_claims) > 0,
        'num_claims': len(validated_claims),
        'claims_json': [c.dict() for c in validated_claims]
    }

async def main(dataset_path: str, output_path: str):
    """Main execution loop with resume capability."""
    df = pd.read_csv(dataset_path)

    for idx, row in df.iterrows():
        for prefix, provider in PROVIDERS.items():
            if not has_result(df, idx, prefix):
                result = await run_extraction_for_sentence(
                    row['sentence'], provider
                )

```

```

df.at[idx, f'{prefix}_binary_result'] = \
    result['binary_result']
df.to_csv(output_path, index=False)
print(f"Processed {idx+1}/{len(df)}")

```

A.5 Analysis Script

The following script calculates extraction metrics by comparing LLM outputs against BingCheck ground truth.

```

#!/usr/bin/env python3
"""
Script to analyze extraction results from all three LLMs
and determine performance metrics.
"""

import pandas as pd
from sklearn.metrics import (
    accuracy_score, precision_score, recall_score,
    f1_score, confusion_matrix
)

def calculate_extraction_metrics(df: pd.DataFrame,
                               provider_prefix: str) -> dict:
    """
    Calculate extraction metrics for a specific LLM provider.
    """
    # Ground truth column
    y_true = df['contains_factual_claim'].values

    # Model predictions
    y_pred = df[f'{provider_prefix}_binary_result'].values

    # Calculate metrics
    metrics = {
        'accuracy': accuracy_score(y_true, y_pred),
        'precision': precision_score(y_true, y_pred),
        'recall': recall_score(y_true, y_pred),
        'f1_score': f1_score(y_true, y_pred),
    }

```

```

# Confusion matrix
tn, fp, fn, tp = confusion_matrix(y_true, y_pred).ravel()
metrics.update({'tp': tp, 'tn': tn, 'fp': fp, 'fn': fn})

# Negative class metrics
metrics['specificity'] = tn / (tn + fp) if (tn + fp) > 0 else 0
metrics['npv'] = tn / (tn + fn) if (tn + fn) > 0 else 0

return metrics

def main(dataset_path: str, output_path: str):
    """Analyze all providers and save metrics."""
    df = pd.read_csv(dataset_path)

    results = []
    for provider in ['gpt4', 'gemini', 'deepseek']:
        metrics = calculate_extraction_metrics(df, provider)
        metrics['provider'] = provider
        results.append(metrics)

    pd.DataFrame(results).to_csv(output_path, index=False)
    print(f"Metrics saved to {output_path}")

```

A.6 Dataset Sample

Table A.7 presents representative examples from the experimental dataset, illustrating the range of sentence types and ground truth labels. All sentences are taken directly from the BingCheck-derived corpus used in the experiments.

A.6.1 Phase 1: Claim Extraction Sample

Table A.8 presents a representative example from the extraction phase dataset. This row shows how the three LLMs extract claims from a single factual sentence about the Argentine tango. The example demonstrates the different extraction strategies: gpt-4o-mini identifies 3 claims with balanced precision-recall, while Gemini and DeepSeek both extract multiple claims with higher selectivity.

Extracted Claims Detail (gpt-4o-mini)

1. The Argentine tango is a dance and musical form

Table A.7: Sample Sentences from Experimental Dataset (BingCheck-derived)

Sentence	GT
“The Argentine tango is a dance and musical form that originated in Buenos Aires in the late 19th century, and has since become a revered art form worldwide.”	True
“Epigenetics is the study of how the environment and other factors can change the way that genes are expressed.”	True
“However, they have few predators due to their thick skin and large size.”	True
“Improving sleep quality is a common goal for many people who want to enhance their health and well-being.”	False
“Together, we can make a difference for our oceans and our planet.”	False
“Each city has its own challenges and opportunities, but also can learn from the experiences and best practices of others.”	False

GT = Ground Truth (True = contains factual claim, False = no factual claim)

Table A.8: Extraction Phase Sample: Argentine Tango Sentence (From my_thesis_dataset_run1.csv)

Field	Value
Original Sentence	“The Argentine tango is a dance and musical form that originated in Buenos Aires in the late 19th century, and has since become a revered art form worldwide.”
Ground Truth (Factual?)	True
gpt-4o-mini Claims #	3
gpt-4o-mini Binary	True
gemini-2.5-flash Claims #	5
gemini-2.5-flash Binary	True
deepseek-v3.2 Claims #	5
deepseek-v3.2 Binary	True

2. The Argentine tango originated in Buenos Aires
3. The Argentine tango originated in the late 19th century

Extracted Claims Detail (gemini-2.5-flash)

1. The Argentine tango is a dance
2. The Argentine tango is a musical form
3. The Argentine tango originated in Buenos Aires
4. The Argentine tango originated in the late 19th century
5. The Argentine tango has become an art form worldwide

Extracted Claims Detail (deepseek-v3.2)

1. The Argentine tango is a dance form
2. The Argentine tango is a musical form
3. The Argentine tango originated in Buenos Aires
4. The Argentine tango originated in the late 19th century
5. The Argentine tango has become a revered art form worldwide

A.6.2 Phase 2: Claim Verification Sample

Table A.9 presents a representative example from the verification phase. This claim about the Argentine tango was extracted in Phase 1 and then evaluated against web search results. The example demonstrates how all three models reached the same verdict (Supported) by finding multiple authoritative sources (UNESCO, Wikipedia, academic databases).

Table A.9: Verification Phase Sample: Argentine Tango Claim (From my_thesis_benchmark_claims_run1.csv)

Claim ID	B001_C00
Claim Text	“The Argentine tango is a dance and musical form”
Original Sentence	“The Argentine tango is a dance and musical form that originated in Buenos Aires in the late 19th century, and has since become a revered art form worldwide.”
Ground Truth Verdict	Supported
Ground Truth Source	https://ich.unesco.org/en/RL/tango-00258
gpt-4o-mini Verdict	Supported
gemini-2.5-flash Verdict	Supported
deepseek-v3.2 Verdict	Supported
Inter-Model Agreement	100% (unanimous)

gpt-4o-mini Reasoning

“The claim that the Argentine tango is a dance and musical form is directly confirmed by multiple reliable sources. Source 2 explicitly states that Argentine tango is both a musical genre and a social dance, and Source 1 indicates that tango is a dance with musical origins. No contradictory evidence is presented.”

gpt-4o-mini Retrieved Sources

1. **Tango - Wikipedia:** “Tango is a dance that has influences ... dance originated in working-class districts of Buenos Aires and Montevideo. Tango music derived from the fusion of various forms of music from Europe...”

2. **Argentine tango - Wikipedia:** “Argentine tango is a musical genre and accompanying social dance originating at the end of the 19th century in the suburbs of Buenos Aires. It typically has a 2/4 or 4/4 rhythmic time signature...”
3. **Guide to Tango Music - MasterClass:** “A tango orchestra, or orquesta ... well as vocalists and tango dancers. The history of tango dates back to the nineteenth century...”

gemini-2.5-flash Reasoning

“Multiple reliable sources explicitly state that Argentine tango is both a dance and a musical form or musical genre.”

gemini-2.5-flash Retrieved Sources

1. **Tango - Wikipedia:** “Tango is a dance that has influences ... dance originated in working-class districts of Buenos Aires and Montevideo. Tango music derived from the fusion of various forms of music from Europe...”
2. **A Short History Of Argentine Tango - Tango.ORG:** “Argentine Tango as a dance and musical form developed in Buenos Aires at the end of the 19th Century, and arrived in Europe and then the US in the first decades of the 1900s. This is the mode of dance that evolved into Ballroom tango.”
3. **Argentine tango - Wikipedia:** “Argentine tango is a musical genre and accompanying social dance originating at the end of the 19th century in the suburbs of Buenos Aires. It typically has a 2/4 or 4/4 rhythmic time signature...”

deepseek-v3.2 Reasoning

“All three sources directly confirm the claim that Argentine tango is a dance and musical form. Source 1 describes it as a partner dance and social Latin dance, Source 2 explicitly states it as a dance and musical form, and Source 3 discusses its evolution as a dance, collectively providing authoritative and consistent evidence.”

deepseek-v3.2 Retrieved Sources

1. **Tango - Wikipedia:** “Tango is a partner dance and social Latin dance that originated in the 1880s along the Río de la Plata, the natural border between Argentina and Uruguay. The tango was born in the impoverished port areas of these countries from a combination of Argentine Milonga, Spanish-Cuban Habanera, and...”

2. **A Short History Of Argentine Tango - Tango.ORG:** “Argentine Tango as a dance and musical form developed in Buenos Aires at the end of the 19th Century, and arrived in Europe and then the US in the first decades of the 1900s. This is the mode of dance that evolved into Ballroom tango.”
3. **Tango | Argentine, Latin American, Couple - Britannica:** “The tango evolved about 1880 in dance halls and perhaps brothels in the lower-class districts of Buenos Aires, where the Spanish tango, a light-spirited variety of flamenco, merged with the milonga, a fast, sensual, and disreputable Argentine...”

Bibliography

- [BharathxD, 2024] BharathxD (2024). ClaimeAI: Open source fact-checking pipeline. <https://github.com/BharathxD/ClaimeAI>. (Accessed: 2025-10-15). The baseline architecture adapted for the TruthLens framework.
- [Brave Software, 2025] Brave Software (2025). Brave search API. <https://api-dashboard.search.brave.com/app/documentation/>. (Accessed: 2025-12-26).
- [Cheng et al., 2024] Cheng, J., Marone, M., Weller, O., Lawrie, D., Khashabi, D., and Van Durme, B. (2024). Dated data: Tracing knowledge cutoffs in large language models. *arXiv preprint arXiv:2403.12958*. Available at: <https://arxiv.org/abs/2403.12958> (Accessed: 2025-10-25). Analysis of static knowledge limitations in LLMs.
- [Colvin et al., 2024] Colvin, S. et al. (2024). Pydantic: Data validation using python type hints. <https://docs.pydantic.dev/>. (Accessed: 2025-10-28). Data validation library used for structured output enforcement.
- [DeepSeek-AI, 2025] DeepSeek-AI (2025). DeepSeek-V3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*. Available at: <https://arxiv.org/abs/2512.02556> (Accessed: 2025-12-28). Technical Report for DeepSeek-V3.2.
- [Dhuliawala et al., 2024] Dhuliawala, S., Komeili, M., Xu, J., Raileanu, R., Li, X., Celikyilmaz, A., and Weston, J. (2024). Chain-of-verification reduces hallucination in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3563–3578. Chain-of-Verification (CoVe) method for self-correction in LLMs.
- [Diggelmann et al., 2020] Diggelmann, T., Boyd-Graber, J., Bulian, J., Ciaramita, M., and Leippold, M. (2020). CLIMATE-FEVER: A dataset for verification of real-world climate claims. 1,535 real-world climate claims adapted from FEVER methodology.

- [Gamma et al., 1994] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley. The definitive reference for the Strategy Pattern.
- [Gemini Team, Google, 2025] Gemini Team, Google (2025). Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*. Available at: <https://arxiv.org/abs/2507.06261> (Accessed: 2025-12-05). Technical Report for Gemini 2.5 Flash.
- [Guo et al., 2022] Guo, Z., Schlichtkrull, M., and Vlachos, A. (2022). A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206. Comprehensive survey unifying definitions and cataloging datasets for automated fact-checking.
- [Gupta and Srikumar, 2021] Gupta, A. and Srikumar, V. (2021). X-fact: A new benchmark dataset for multilingual fact checking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 675–682, Online. Association for Computational Linguistics. Largest multilingual fact-checking dataset with 25 languages.
- [Hu et al., 2022] Hu, X., Guo, Z., Wu, G., Liu, A., Wen, L., and Yu, P. (2022). CHEF: A pilot Chinese dataset for evidence-based fact-checking. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3362–3376, Seattle, United States. Association for Computational Linguistics. First Chinese evidence-based fact-checking dataset with 10K claims.
- [Huang et al., 2024] Huang, L., Yu, W., Ma, W., et al. (2024). A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 1(1). Defines Factuality vs. Faithfulness Hallucinations.
- [Ji et al., 2023] Ji, Z. et al. (2023). Survey of hallucination in large language models. *ACM Computing Surveys*, 55(12). Standard survey paper defining Hallucinations (Intrinsic/Extrinsic).
- [Jurafsky and Martin, 2025] Jurafsky, D. and Martin, J. H. (2025). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, with Language Models*. 3rd edition. Online manuscript released August 24, 2025 (URL accessed: 2025-12-26).

- [Landis and Koch, 1977] Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174.
- [LangChain AI, 2024] LangChain AI (2024). LangGraph: Building stateful multi-actor applications with LLMs. <https://langchain-ai.github.io/langgraph/>. (Accessed: 2025-11-02). Graph-based orchestration framework for agentic workflows.
- [Li et al., 2024] Li, M., Peng, B., Galley, M., Gao, J., and Zhang, Z. (2024). Self-checker: Plug-and-play modules for fact-checking with large language models. *Findings of the Association for Computational Linguistics: NAACL 2024*. Available at: <https://aclanthology.org/2024.findings-naacl.156/> (Accessed: 2025-10-20). Original creators of the BingCheck dataset.
- [Liu et al., 2025] Liu, F., Abuadbbba, A., Moore, K., Nepal, S., Paris, C., Wu, J., Yang, J., and Sheng, Q. Z. (2025). Adversarial attacks against automated fact-checking: A survey. Survey on adversarial vulnerabilities in fact-checking systems, accepted to EMNLP 2025.
- [Metropolitansky and Larson, 2025] Metropolitansky, D. and Larson, J. (2025). Towards effective extraction and evaluation of factual claims. *arXiv preprint arXiv:2502.10855*. Available at: <https://arxiv.org/abs/2502.10855> (Accessed: 2025-11-05). The Claimify Paper.
- [Open Source Initiative, 2024] Open Source Initiative (2024). The open source ai definition 1.0. <https://opensource.org/ai/open-source-ai-definition>. (Accessed: 2025-12-26).
- [OpenAI, 2024] OpenAI (2024). GPT-4o mini: Advancing cost-efficient intelligence. <https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/>. (Accessed: 2025-12-09).
- [Pan et al., 2024] Pan, R., Cao, B., Lin, H., Han, X., Zheng, J., Wang, S., Cai, X., and Sun, L. (2024). Not all contexts are equal: Teaching LLMs credibility-aware generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 19873–19889. Credibility-aware Generation (CAG) for evidence quality assessment.
- [Saakyan et al., 2021] Saakyan, A., Chakrabarty, T., and Muresan, S. (2021). COVID-fact: Fact extraction and verification of real-world claims on COVID-19 pandemic. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2116–2129, Online. Associa-

tion for Computational Linguistics. Dataset of 4,086 COVID-19 claims for health misinformation detection.

- [Thorne et al., 2018] Thorne, J., Vlachos, A., Christodoulopoulos, C., and Mittal, A. (2018). FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Standard citation for traditional fact-checking pipelines.
- [Varshney et al., 2023] Varshney, N., Yao, W., Zhang, H., Chen, J., and Yu, D. (2023). A stitch in time saves nine: Detecting and mitigating hallucinations of LLMs by validating low-confidence generation. *arXiv preprint arXiv:2307.03987*. Available at: <https://arxiv.org/abs/2307.03987> (Accessed: 2025-11-10). Confidence-based hallucination detection.
- [Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- [Wadden et al., 2022] Wadden, D., Lo, K., Kuehl, B., Cohan, A., Beltagy, I., Wang, L. L., and Hajishirzi, H. (2022). SciFact-Open: Towards open-domain scientific claim verification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4719–4734, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. Large-scale scientific claim verification over 500K research abstracts.
- [Wang et al., 2024] Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., et al. (2024). A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345. Defines the unified framework for LLM-based agents.
- [Wei et al., 2024] Wei, J., Huang, J., Yang, C., et al. (2024). Long-form factuality in large language models. *arXiv preprint arXiv:2403.18802*. Available at: <https://arxiv.org/abs/2403.18802> (Accessed: 2025-10-12). The SAFE Framework paper.
- [Zheng et al., 2023] Zheng, L., Chiang, W.-L., Sheng, Y., et al. (2023). Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36. Standard citation for the LLM-as-a-Judge concept.