Universitat Oberta de Catalunya (UOC)

Master's Degree in Data Science

MASTER'S THESIS

Area: Natural Language Processing

Factible: A Multi-Agent System for Automated Fact-Checking of YouTube Videos

---

Author: Begoña Echavarren Sánchez

Tutor: Josep-Anton Mir Tutusaus

December 2025

Copyright

Source Code:

The complete source code for this project is publicly available at:

`https://github.com/begoechavarren/factible`

FINAL PROJECT RECORD

Title of the project:	Factible: A Multi-Agent System for Automated Fact-Checking of YouTube Videos
Author's name:	Begoña Echavarren Sánchez
Tutor's name:	Josep-Anton Mir Tutusaus
Delivery date:	12/2025
Degree or program:	Master's Degree in Data Science
Final Project area:	Natural Language Processing
Language of the project:	English
Keywords:	fact-checking, LLM, multi-agent systems

Acknowledgements

I would like to thank my tutor, Josep-Anton Mir Tutusaus, for his support since the very beginning of this project. His warm reception of the idea, continuous encouragement, and genuine enthusiasm were truly appreciated. His expertise and insightful feedback definitely helped shape this work in a meaninful way.

I would also like to thank my family and friends for their patience, encouragement, and understanding—not only during these past months, but throughout my entire master's studies. Their continuous support has been essential in helping me develop the skills that made this thesis possible.

Abstract

Misinformation on video platforms represents a growing challenge in the digital age. YouTube, with over 2 billion monthly users and 500 hours of video uploaded every minute, serves as a primary information source for millions worldwide. However, the absence of effective verification mechanisms allows false or misleading claims to spread at massive scale, impacting public health, democratic processes, and social cohesion.

This thesis presents Factible, a multi-agent system for automated fact-checking of YouTube videos. The system implements an end-to-end pipeline that processes video content through five specialized components: transcript extraction, claim detection with importance-based ranking, query generation for evidence retrieval, open-web evidence collection with source reliability assessment, and verdict synthesis with confidence levels.

The implementation leverages large language models (LLMs) for reasoning tasks while employing classical algorithms for deterministic operations. Key innovations include a novel approach for claim importance ranking, adaptive credibility filtering for source quality, and a three-level parallelization architecture for latency optimization.

Evaluation on 30 annotated YouTube videos across health, climate, and political topics demonstrates strong performance: 81.3% claim extraction precision, 94.7% evidence retrieval success rate, and 73.3% verdict accuracy. The system achieves practical efficiency with mean processing time of 129.6 seconds per video at $0.003 average cost, enabling cost-effective deployment for individual users.

The complete system, including a web-based interface with real-time streaming, is publicly available as open-source software, bridging the gap between academic research and practical fact-checking tools accessible to non-expert users.

Keywords: automated fact-checking, large language models, multi-agent systems, natural language processing, misinformation detection, YouTube, information retrieval, evidence-based verification

Contents

List of Figures

List of Tables

1 Introduction

Misinformation on digital platforms has become a major challenge in the information age. YouTube, with over 2 billion monthly active users, has evolved into a primary information source for millions of people worldwide. However, the absence of effective verification mechanisms allows false or misleading claims to propagate at massive scale, impacting public health, democratic processes, and social cohesion [2, 41].

This thesis addresses the critical need for automated fact-checking systems capable of processing video content from platforms like YouTube. The project develops Factible, a multi-agent system that implements an end-to-end pipeline for extracting, verifying, and synthesizing factual claims from YouTube videos using large language models (LLMs) and web-based evidence retrieval.

1.1 Context and Motivation

1.1.1 The Problem of Video Misinformation

The central problem this project addresses is the **lack of automated, accessible mechanisms to verify factual claims in audiovisual content from digital platforms**, specifically YouTube videos. Individual users lack tools to systematically verify claims, while manual verification requires advanced skills, considerable time, and resources that are not always available.

YouTube presents unique challenges for fact-checking:

- **Scale**: Over 500 hours of video are uploaded every minute, making human-only verification impossible.
- **Passive consumption**: Videos are consumed passively, with viewers less likely to critically evaluate claims than when reading text.
- **Unstructured transcripts**: Video transcripts lack punctuation, structural markers, and clear claim boundaries, with statements embedded in continuous narrative flow.
- **Verification burden**: Manually verifying claims requires searching multiple sources, evaluating their reliability, and synthesizing potentially conflicting information—skills and time most viewers lack.

1.1.2 Relevance and Importance

This project is relevant for multiple reasons:

- **Social impact**: Misinformation erodes trust in institutions, influences political decisions, and affects public health [2, 41]. An automated verification system can contribute to mitigating these effects by enabling faster, more accessible fact-checking.
- **Technological challenge**: The project integrates multiple areas of data science research: natural language processing, information retrieval, source credibility evaluation, and automated reasoning. It represents a technically complex problem requiring innovative solutions.
- **Scalability necessity**: The volume of content generated daily makes purely human fact-checking infeasible. Automation is necessary to address the problem at scale, serving as a force multiplier for human fact-checkers.
- **Media literacy**: Tools that empower citizens to critically evaluate information promote critical thinking and strengthen media literacy in society, contributing to a healthier information ecosystem.

1.1.3 Current State of Solutions

Currently, three main approaches exist for fact-checking:

1. **Manual fact-checking**: Organizations like Newtral, Maldita.es, PolitiFact, and FactCheck.org employ specialized journalists. This approach guarantees quality but is limited by human resources and is inherently slow, typically checking only a few claims per day. Furthermore, manual fact-checking is subject to cognitive biases from individual analysts and potential institutional biases from the organizations themselves, as fact-checking effectiveness is contingent upon perceived trust in the source delivering fact-checks [23].
2. **Academic systems**: Projects like FEVER [37], FEVEROUS [1], and recent work on scientific verification [40] operate primarily on structured knowledge bases (Wikipedia) or require specifically curated datasets. They are not available as products usable by end users and have limited scope.
3. **Verification APIs**: Google Fact Check Tools API [10] aggregates fact-checks already published by professional organizations but only covers content previously verified manually. It does not provide new verification for unexamined content.

Identified gap: None of these approaches provides automated end-to-end verification for continuously generated audiovisual content on platforms like YouTube. A gap exists between academic research (static datasets, limited domains) and practical, accessible tools for end users. Section 2 provides a detailed analysis of existing systems and their limitations.

1.2 Project Objectives

1.2.1 Main Objective

Design, implement, and evaluate a multi-agent system based on language models capable of automatically verifying factual claims in YouTube videos through retrieval and analysis of web-based evidence, generating substantiated verdicts with confidence estimates that are useful for end users.

1.2.2 Specific Objectives

The project pursues six specific objectives:

1. **Multi-agent architecture design**: Define a modular processing pipeline with specialized components, structured data schemas, and an orchestration system for agent coordination.

2. **System component implementation**: Develop five core modules—transcript extraction, claim identification with importance scoring, search query generation, web scraping with source reliability evaluation, and explainable verdict generation.

3. **LLM usage optimization**: Analyze trade-offs between cost, latency, and output quality; evaluate prompt optimization strategies and structured outputs.

4. **System evaluation**: Define technical performance metrics, incorporate LLM-as-a-judge and manual review, and conduct case studies across different video topics.

5. **End-to-end system implementation**: Develop a REST API with real-time streaming, create an intuitive web interface for end users, and ensure reproducibility through documentation.

6. **Ethical and bias considerations**: Address potential biases in sources and models through transparency and multi-perspective evidence presentation, identify limitations of the automated approach, and define best practices for responsible deployment.

1.2.3 Project Scope

Includes:

- YouTube videos with available transcripts (automatic or manual subtitles)
- Factual claims verifiable through public web sources

- Content in English
- On-demand processing of individual videos

Does not include:

- Self-generation of transcripts from audio (speech-to-text)
- Verification of visual content
- Large-scale batch verification
- Content from platforms other than YouTube

1.3 Ethical Considerations and Social Impact

1.3.1 Ethical Commitment and Global Competence

The project addresses three dimensions of ethical and global competence:

(I) Sustainability: The use of LLMs implies significant energy consumption. This is addressed through selection of cost-effective models, support for local inference to avoid cloud dependency, prompt optimization to minimize processed tokens, strict token budgets per component, and transparent documentation of resource consumption.

(II) Ethical behavior and social responsibility: LLMs can perpetuate biases present in their training data. Mitigation strategies include:

- Total transparency in consulted sources with verifiable URLs
- Presentation of evidence showing different stances (supports/refutes/mixed/unclear)
- Explicit evaluation of confidence level in each verdict
- Clear warnings about limitations of the automated system

The system positions itself as a support tool that automates preliminary steps, allowing professionals to focus on complex analyses requiring human judgment.

(III) Diversity and human rights: The system seeks to evaluate diversity of perspectives when multiple web sources with different approaches are available, documenting this limitation when not possible. It does not censor content but provides additional context through verification, respecting freedom of expression. It does not process personal data or identifiable user information.

1.3.2 Sustainable Development Goals (SDGs)

The project contributes to three SDGs from the 2030 Agenda:

SDG 4 (Quality Education): The system acts as a media literacy tool that promotes critical thinking and the ability to evaluate information sources. It facilitates learning about fact-checking, evidence analysis, and recognition of verifiable claims—fundamental competencies in today's digital society.

SDG 16 (Peace, Justice and Strong Institutions): It combats misinformation that erodes trust in democratic institutions and media. It provides transparent, traceable, and accessible verification mechanisms that allow citizens to systematically contrast factual claims, contributing to a healthier, more resilient information ecosystem.

SDG 10 (Reduced Inequalities): It democratizes access to fact-checking through a free, open-source tool, reducing the gap between professional organizations with specialized resources (fact-checkers, journalists) and individual citizens. It empowers users without specialized training to critically evaluate content on digital platforms.

1.4 Methodology

1.4.1 Research Strategy

This project adopts a **Design and Creation** strategy [28], appropriate for information systems research where a new technological artifact is developed whose construction and evaluation constitute the main contribution to knowledge. According to Hevner et al. [14], this strategy requires: (1) creating a viable artifact, (2) addressing a relevant problem, (3) conducting rigorous evaluation, (4) making clear contributions, (5) applying rigor in construction, (6) designing as a search process, and (7) effectively communicating results.

Data collection techniques for evaluation:

Quantitative:

- Automated measurement of system metrics: processing time, cost (tokens × price), number of retrieved sources, stance distribution
- Evaluation using LLM-as-a-judge to assess quality of extracted claims and generated verdicts
- System efficiency metrics: end-to-end latency, evidence retrieval success rate

Qualitative:

- Manual review of system outputs (claim relevance, verdict coherence)
- Detailed case studies across different topics

1.4.2 Development Methodology

Chosen strategy: Development of a new end-to-end product with a functional web interface.

Justification: This strategy is most appropriate because it allows evaluation of the complete practical viability of the system under real usage conditions, not just as a theoretical concept. Modular development from scratch facilitates experimentation with different prompt strategies and parameter configurations, and guarantees total control over the verification pipeline. Including a web interface demonstrates applicability for end users and allows collecting feedback on real system usability.

Iterative development process:

1. Implementation of individual components with isolated testing
2. End-to-end pipeline integration (functional MVP)
3. Iterative refinement module by module based on results
4. Systematic evaluation against ground truth annotations

1.5 Summary of Project Outputs

This thesis produces the following outputs:

1. **Factible**: A complete, open-source multi-agent fact-checking system for YouTube videos, available at https://github.com/begoechavarren/factible
2. **Modular pipeline architecture**: Five specialized components (Transcriptor, Claim Extractor, Query Generator, Online Search, Output Generator) with well-defined interfaces
3. **Web interface**: React-based frontend with real-time SSE streaming for interactive verification
4. **REST API**: FastAPI-based backend enabling programmatic access to fact-checking functionality
5. **Evaluation framework**: Infrastructure for systematic experimentation including tracking, metrics computation, and result analysis
6. **Annotated evaluation dataset**: 30 YouTube videos with 503 ground truth claims across health, climate, and political topics
7. **This thesis document**: Comprehensive documentation of the design, implementation, and evaluation of the system

2 State of the Art

Automated fact-checking has evolved significantly since the mid-2010s, progressing from rule-based systems through classical machine learning classifiers and knowledge graph approaches, to neural architectures and contemporary Large Language Models (LLMs). This evolution has been driven by advances in natural language processing, retrieval-augmented generation (RAG), and multi-agent systems that break down fact-checking into specialized, coordinated tasks.

This section examines automated fact-checking systems, emphasizing multi-agent architectures for video content verification. It analyzes the technological foundations of each pipeline component, from claim detection to evidence retrieval to verdict synthesis, and identifies gaps limiting practical deployment, tracing the field's evolution from 2014 to the present.

2.1 Historical Development of Automated Fact-Checking

Automated fact-checking emerged as a formal NLP task in the mid-2010s. Vlachos and Riedel [39] provided the foundational task definition, framing fact-checking as a pipeline comprising claim identification, evidence retrieval, and veracity prediction—a structure that remains central to contemporary systems.

Early approaches (2014–2017) relied on two main paradigms. **Knowledge graph verification** systems, such as those developed by Ciampaglia et al. [6], computed shortest paths through structured knowledge bases like DBpedia to assess claim truthfulness, achieving promising results on well-formed factual statements but struggling with natural language variability. **Classical machine learning classifiers**, exemplified by ClaimBuster [12], used SVMs and Random Forests with hand-crafted features including TF-IDF vectors, named entity counts, part-of-speech patterns, and sentiment scores to identify check-worthy claims in political discourse.

The release of benchmark datasets marked a turning point. The LIAR dataset (2017) provided 12,836 PolitiFact claims with six-way veracity labels, while **FEVER** [37], with its 185,445 claims verified against Wikipedia, established the dominant evaluation paradigm and formalized the retrieve-then-verify pattern. These datasets enabled systematic comparison of approaches and drove rapid progress.

Pre-transformer neural models (2016–2018) achieved substantial improvements over classical approaches. The Decomposable Attention Model [31] and ESIM introduced attention mechanisms for natural language inference, achieving approximately 70% accuracy on FEVER. However, the introduction of BERT in late 2018 quickly dominated leaderboards, demonstrating that transfer learning from large-scale pretraining could surpass task-specific architectures.

Despite this progress, pre-LLM systems exhibited fundamental limitations that motivate current research directions: (1) **single-source restriction**—systems verified against only Wikipedia or curated knowledge bases, not diverse web sources; (2) **no explanation generation**—models predicted labels without human-readable justifications; (3) **binary/ternary labels**—poor handling of nuanced "partially true" claims; (4) **no multimodal capability**—text-only processing, with video/audio fact-checking treated as a separate field; and (5) **weak multi-hop reasoning**—claims requiring evidence synthesis across multiple documents frequently failed. These limitations directly inform the design of modern multi-agent LLM architectures, which aim to address each gap through specialized agents, web-scale retrieval, and natural language reasoning.

2.2 Multi-Agent Architectures for Fact-Checking

2.2.1 Foundational Multi-Agent Frameworks

FactAgent [45] introduced an agentic workflow that explicitly emulates the methodology of human fact-checkers. Instead of fine-tuning models for specific tasks, FactAgent employs a pre-trained LLM that operates through a structured script: (1) gathering evidence via tools, (2) analysing that evidence, and (3) synthesising a verdict. A key advantage of this approach is its zero-shot nature: the system uses the LLM's existing capabilities without requiring task-specific training data. However, the sequential multi-step process can introduce latency, and errors in early stages may cascade through the pipeline.

LoCal [3] handles complex claims through a decomposition–reasoning–evaluation loop. It uses specialized agents: a decomposer breaking claims into sub-claims, reasoning agents verifying each with evidence, and evaluators ensuring logical consistency and testing counterfactual scenarios. If inconsistencies arise, agents iteratively revise their reasoning. This targets single-pass verification weaknesses but increases computational cost.

Multi-agent debate systems [16, 20] use adversarial collaboration where distinct LLM agents take opposing roles: one argues for a claim's truth, another against it, while a judge decides the outcome. This exposes contradictions and forces agents to defend positions with evidence, reducing confirmation bias and hallucinations. The adversarial setup prevents premature convergence on incorrect conclusions, though debates can reach impasses and quality depends on the judge agent's synthesis ability.

2.2.2 Evolution and Current Landscape

Multi-agent design for fact-checking emerged recently (2023–2025). Early research used linear pipelines where a single model performed one task at a time. By 2024, works like FactAgent

and LoCal began breaking verification into specialized agents [3, 45]. In 2025, researchers added debate mechanisms, self-reflection, and richer tool use [15, 20, 38]. MAD-Sherlock showed that debate-driven systems reduce hallucinations through collaborative verification [16]. Despite progress, recent evaluations reveal latency challenges, high compute requirements, and the need for careful orchestration to avoid loops or premature termination.

2.3 Automatic Claim Detection in Text

Identifying which statements warrant fact-checking—known as check-worthiness detection—is crucial for any verification pipeline. For YouTube videos, this means scanning transcripts to flag claims that are both verifiable and important.

2.3.1 Traditional Supervised Approaches

Claim detection research began in the mid-2010s, targeting political speech. **ClaimBuster** [12] pioneered the approach, using supervised models trained on human-labelled debate transcripts to score sentences for verifiable factual claims. Subsequent work refined datasets and models: **ClaimRank** introduced context-aware modeling using surrounding sentences [9], while the **CLEF CheckThat!** lab (2018–present) released multilingual challenges for check-worthiness detection [7]. These efforts established claim detection as a classification problem, with BERT and transformers becoming dominant after 2018. The key insight was that check-worthiness requires assessing both **importance** and **verifiability**—a claim like "Unemployment rose by 15% last quarter" is check-worthy because it is both verifiable and consequential.

2.3.2 LLM-Based Claim Detection

Recent work explores whether LLMs can identify check-worthy claims without fine-tuning. Sawiński et al. [34] compared fine-tuned BERT variants with GPT-3/GPT-4 in zero-shot mode, finding that simple prompts still underperform fine-tuned models due to inconsistent internal definitions of "worthiness" and sensitivity to prompt wording.

However, careful prompt design can substantially improve LLM performance. Li et al. [18] demonstrated that LLMs with verbose few-shot prompts can effectively replace traditional claim detectors in automated pipelines. Ni and Carter [25] proposed a three-step prompting approach where the LLM analyzes text in stages—highlighting factual statements, applying check-worthiness criteria, and ranking by importance—similar to chain-of-thought reasoning [26].

2.3.3 Current Challenges

Key challenges include: (1) **definition ambiguity**: what constitutes a "check-worthy" claim varies by context; (2) **scalability**: scanning long transcripts with LLMs is slow and expensive; (3) **false positives**: overly aggressive detection wastes resources; and (4) **domain adaptation**: models trained on political debates may not work for scientific or economic content.

2.4 RAG Approaches in Automated Fact-Checking

A core pillar for automated fact-checking is retrieval-augmented generation (RAG), which combines text generation with external information retrieval to ensure outputs are based on verifiable sources.

2.4.1 From RAG to Fact-Checking

Lewis et al. [17] formalized RAG for general knowledge-intensive tasks, showing that augmenting generation with external knowledge retrieval significantly improves performance. For fact-checking, this paradigm is essential: systems must fetch reliable sources that support or refute claims. The **FEVER** dataset exemplified this retrieve-then-verify pattern: given a claim, retrieve relevant documents (e.g., Wikipedia pages), then determine if they support or refute the claim [37]. Modern systems use LLMs for both retrieval and verification, conditioning their reasoning on retrieved evidence.

RAG addresses a critical limitation of pure LLM approaches: parametric knowledge can be outdated, incomplete, or hallucinated. By retrieving external information (from document indexes, web search, or APIs), RAG systems access current information, provide source attribution, and ground reasoning in verifiable evidence. This is crucial for fact-checking, where claims often reference recent events, specific statistics, or specialized knowledge not in LLM training data.

2.4.2 Tool Use and Web Retrieval

Integrating tool use with LLMs has enabled more sophisticated retrieval. The **ReAct pattern** (Reason and Act) interleaves tool use with chain-of-thought reasoning, letting LLMs decide when to call external tools like search engines [44]. **WebGPT** demonstrated training LLMs to use web browsers to answer questions and cite sources, improving factual accuracy by teaching models to navigate search results and synthesize information from multiple pages [24]. Similarly, **Toolformer** showed that LLMs can be fine-tuned to call external tools like search engines or calculators, reducing hallucinations by grounding answers in retrieved evidence [35].

Recent work has applied these techniques to fact-checking. **Chern et al. (2023)** proposed using Google Search, Google Scholar, and other tools to verify LLM-generated text against external sources [4], while **Cheung and Lam (2023)** combined search-engine retrieval with LLaMA to predict claim veracity [5]. These tool-augmented methods address LLMs' inherent knowledge limitations, which can be outdated or incomplete [4].

2.4.3 Open Web versus Closed Knowledge Bases

Most academic fact-checking systems restrict retrieval to trusted corpora (primarily Wikipedia) to simplify evaluation and ensure evidence quality. This yields high precision but severely limits real-world coverage [8]. The FEVER dataset, while influential, exemplifies this by assuming all claims can be checked against a June 2017 Wikipedia snapshot, which breaks down for recent events, specialized domains, or claims requiring sources like World Bank reports or CDC guidelines.

Tian et al. (2024) integrated web-retrieval agents into an LLM pipeline and demonstrated improved misinformation detection [38]. However, open-web retrieval introduces challenges: (1) **source credibility**: not all websites are reliable; (2) **information quality**: web content varies in accuracy; (3) **ranking complexity**: identifying relevant sources among millions of candidates; and (4) **dynamic nature**: content changes, affecting reproducibility.

Current best practices include prioritizing sources with high domain authority (established news organizations, academic institutions, government agencies), cross-referencing multiple independent sources, explicitly evaluating source credibility using metadata (publication date, author credentials, institutional affiliation), and maintaining transparency by exposing retrieved sources to users. Systems like FactAgent incorporate evidence retrieval as a dedicated step, using search tools to query the web and filtering results based on relevance and credibility [45].

2.4.4 Query Optimization for Fact-Checking

A critical but often overlooked component of RAG systems is query formulation—the same claim can be verified or refuted depending on how search queries are constructed [20]. Effective strategies include **keyword extraction** (identifying salient terms and entities), **query expansion** (generating variants to capture different phrasings, e.g., "unemployment rate increased" alongside "jobless claims rose") [17], and **temporal awareness** (incorporating time constraints for period-specific claims).

Recent multi-agent systems often dedicate a specialized agent to query generation, recognizing that poor queries degrade overall performance regardless of verification model quality. FactAgent includes explicit query formulation as one of its agent steps, using the LLM to generate search-optimized queries [45].

2.5 LLM-Based Claim Verification Methods

Once claims are identified and evidence retrieved, systems must determine veracity, labelling claims as supported, refuted, mixed, or not enough evidence. Traditional approaches treated this as textual entailment, using classifiers to determine if evidence entails or contradicts claims. With LLMs, a new approach emerged: using models to perform verification through natural language reasoning.

2.5.1 Prompting Strategies

Zero-shot and few-shot prompting involves providing an LLM with a claim and evidence, asking it to decide veracity and explain why: "Claim: X. Evidence: [text]. Based on the evidence, is the claim true or false?" [46]. In zero-shot mode, the LLM relies on internal reasoning and evidence interpretation. In few-shot mode, the prompt includes examples of claims with evidence and the correct verdict to guide the model.

GPT-4 and similar models show surprising capability at this task, often correctly interpreting whether evidence supports statements. However, LLMs can be overly agreeable, sometimes hallucinating justifications or defaulting to "Supported" even when evidence is insufficient [46]. This confirmation bias stems from models' training to be helpful and provide answers, even when saying "I don't know" would be more appropriate.

Careful prompt engineering can mitigate this bias. Effective strategies include explicitly instructing the model to answer "Not Enough Evidence" or "Mixed" when appropriate, adding system messages emphasizing accuracy over helpfulness, requesting citation of specific evidence sentences, using temperature settings near zero to reduce randomness, and implementing multi-pass verification where the model critiques its own reasoning.

2.5.2 Advanced Reasoning Strategies

More sophisticated approaches use **chain-of-thought reasoning** or implement the LLM as an agent in a loop. The **ReAct pattern** has the LLM explicitly reason step-by-step while using tools [44]—for verification, this involves breaking claims into parts, querying search engines, evaluating evidence, and synthesizing conclusions. FactAgent exemplifies this approach: the LLM follows a script where each step is explicit and logged for transparency [45]. Chain-of-thought provides transparency and debuggability, though multiple LLM calls can be slow and errors compound across stages.

Complementary techniques address reliability concerns. **SelfCheckGPT** detects hallucinations by generating multiple independent answers and checking consistency—hallucinated information varies across samples while grounded information remains stable [21]. **LLM-as-**

a-judge approaches use one model to generate answers and another to verify them [32, 33], while **cross-model checking** uses different models that can abstain when they disagree [1]. Many pipelines also incorporate **stance detection** to classify whether evidence supports, refutes, or is neutral toward claims [37]. While these techniques improve reliability, they add computational overhead.

2.5.3 Current Capabilities and Limitations

Carefully prompted LLMs can achieve near state-of-the-art performance on tasks like FEVER [37]. However, they still make mistakes, especially on ambiguous or complex claims requiring specialized knowledge or multi-step reasoning. Designing prompts or agent behaviors to be appropriately skeptical remains important, requiring calibration to avoid being too trusting or too skeptical.

2.6 Evaluation of Fact-Checking Systems

Evaluating automated fact-checking systems requires assessing accuracy, explanation quality, evidence usage, and practical usability.

2.6.1 Accuracy and Retrieval Metrics

Standard classification metrics include accuracy, F1-score, and precision/recall. The FEVER challenge introduced the **FEVER score**, which requires both correct labels and proper evidence, penalizing systems that get labels right without grounding [37]. For evidence retrieval, key metrics include **Recall@k**, **Precision**, and **Mean Average Precision (MAP)** [37]. End-to-end evaluations typically credit systems only when they retrieve human-identified evidence, though multiple valid sources may exist for a claim.

2.6.2 Explanation Quality and Consistency

Explanations should be **faithful** (reflect actual reasoning) and **factually consistent** with evidence. While automatic metrics like BLEU or ROUGE exist, they don't measure factuality well [32]. **LLM-as-a-judge** has become popular for scoring explanation coherence and factuality, though these judges require validation against human assessments [32, 33]. **Stance consistency** checks whether evidence stances align with final verdicts, while systems like LoCal evaluate logical consistency by checking if composed solutions imply claim veracity [3].

2.6.3 Human and Computational Evaluation

Human judgment remains the gold standard, with experts rating verdict correctness and lay users evaluating explanation clarity [20, 41]. For practical deployment, computational efficiency also matters: latency, throughput, and cost metrics are critical for operational systems, though rarely reported in academic papers [38, 45].

2.7 Current Limitations and Research Opportunities

Despite rapid progress, automated fact-checking systems have significant limitations constraining practical deployment.

2.7.1 Lack of End-to-End Usability

Most research prototypes focus on isolated components rather than seamless end-to-end tools. Some excel at claim detection but assume manual verification [12], while others verify claims but require human identification. Even ClaimBuster, dubbed "end-to-end", only highlighted claims without verification [12]. Complete systems need to integrate detection, verification, and source tracing, but existing systems typically address only one or two steps [19]. For YouTube videos, true end-to-end systems should handle transcription extraction, claim detection, evidence retrieval, verification, and user-friendly presentation, but this integration is rarely achieved [19].

2.7.2 Dependence on Structured Sources

Much research restricts evidence to structured knowledge bases, primarily Wikipedia. While this yields cleaner evaluation, it severely limits applicability [1]. Real misinformation often requires specialized sources not available in Wikipedia, and systems benchmarked on FEVER tend to be overfitted [37]. In practice, fact-checkers must handle the open web including news sites, scientific papers, and government databases, introducing challenges of source credibility and information quality [1].

2.7.3 Production Readiness and Scalability

Most solutions remain research-grade implementations rather than production-ready systems. Code is typically provided as research artifacts without the robustness or user interfaces needed for public deployment [19]. Even public tools like Google Fact Check Explorer only search existing fact-checks rather than performing new verification [10]. Scalability poses additional challenges: if verification takes minutes per claim, videos with 20 claims become impractical for

interactive use. Cost is also a factor—using commercial LLMs for every step can be prohibitively expensive at scale [19].

2.7.4 Trust, Transparency, and Model Selection

Users may be reluctant to trust AI verdicts without understanding their derivation, and many systems have been criticized as "black boxes" [41]. While multi-agent systems and chain-of-thought approaches attempt to address this via explicit reasoning traces, they face risks of hallucinated explanations. Additionally, despite many LLM options (commercial models like GPT-4 or Claude; open-source models like LLaMA, Qwen, DeepSeek), there's limited systematic comparison of their suitability for fact-checking tasks—research tends to use whichever model is most accessible without careful comparison of trade-offs [32].

2.7.5 The Video Content Gap

Most fact-checking research focuses exclusively on text-based content, primarily analyzing written articles, social media posts, or pre-extracted claims from political debates [12, 37]. YouTube, despite having over 2 billion monthly active users and serving as a primary information source for millions, remains largely unaddressed by automated fact-checking systems. Existing research assumes pre-processed text, leaving the video-to-claim pipeline unsolved. Video content introduces unique challenges including automatic speech recognition errors, missing punctuation and formatting, temporal context loss, and high computational costs. Given YouTube's central role in information consumption and its documented contribution to misinformation spread [2], this gap represents a critical limitation in automated fact-checking capabilities.

3 Materials and Methods

This section presents the comprehensive technical implementation of Factible, a multi-agent system for automated fact-checking of YouTube videos. The complete source code is publicly available at `https://github.com/begoechavarren/factible`. The system implements an end-to-end pipeline that processes video content through five specialized components, leveraging large language models (LLMs) for reasoning tasks while employing classical algorithms for deterministic operations. The implementation follows design-science principles [14, 28] and focuses on building practical artifacts that are iteratively evaluated and refined.

3.1 System Architecture Overview

3.1.1 High-Level Architecture

Factible implements an end-to-end automated fact-checking pipeline for YouTube videos using a multi-agent architecture. Recent research on LLM agents demonstrates that multi-agent collaboration can enhance factuality and reasoning by allowing specialized agents to coordinate on tasks [42]. FactAgent further shows that decomposing fact-checking into dedicated agents for input ingestion, query generation, evidence retrieval, and verdict prediction yields higher accuracy and transparency [45]. The Factible architecture follows this line of work by processing video content through five specialized, modular components that operate sequentially with three levels of internal parallelization.

The system processes a YouTube video URL through five sequential stages, each with specialized responsibilities. The pipeline begins with transcript extraction, proceeds through claim and query generation, conducts online evidence retrieval, and culminates in structured verdict synthesis. This modular design enables independent optimization of each component while maintaining clear data contracts between stages.

The five stages are:

1. **Transcriptor**: Extracts video transcripts via YouTube Transcript API, preserving timestamped segments for claim localization. The component includes automatic fallback to proxy service when rate-limited, ensuring robust transcript retrieval across different access conditions.

2. **Claim Extractor** (LLM Agent): Infers the video's central argument before extracting factual and verifiable claims. Each claim receives an importance score based on its impact on the video's thesis. Post-processing locates claims within the transcript for timestamp mapping.

3. **Query Generator** (LLM Agent): Generates diverse search queries across four strategic types—direct, alternative, source, and context. Each query receives a priority score (1–5) based on evidence likelihood, enabling filtering of low-priority queries.

4. **Online Search**: Executes a four-step evidence retrieval pipeline for each query: (i) Google Search via Serper API, (ii) website reliability assessment using Media Bias/Fact Check (MBFC) data combined with TLD reputation and domain age heuristics [22], (iii) content fetching via Selenium WebDriver with JavaScript rendering support, and (iv) LLM-based evidence extraction with stance classification (supports, refutes, mixed, unclear).

5. **Output Generator** (LLM Agent): Synthesizes evidence into structured verdicts by building evidence bundles grouped by stance, generating natural language summaries with confidence levels, calculating algorithmic evidence quality scores, and mapping claims to video timestamps for interactive navigation.

Figure 1 illustrates the complete pipeline architecture with data flow and parallelization points across all five stages.

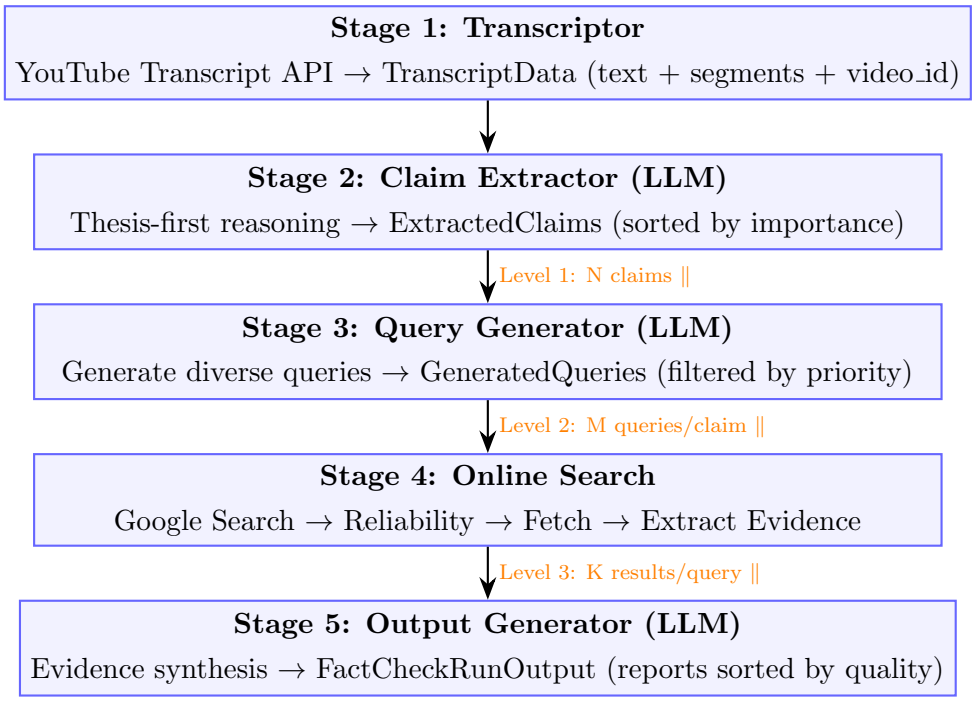


Figure 1: High-level pipeline architecture showing the five main stages and three parallelization levels. Parallelization occurs at claims (Level 1), queries per claim (Level 2), and search results per query (Level 3). Verdict generation happens within Level 1 after each claim's evidence is collected.

3.1.2 Design Principles

The system adheres to several key design principles derived from software engineering best practices and GenAI application development [14]:

1. **Modularity**: Each component is isolated with well-defined inputs and outputs using Pydantic schemas, enabling independent optimization, testing, and replacement.

2. **Structured Outputs**: All LLM interactions use Pydantic AI with typed output schemas, ensuring type safety, automatic validation, and consistent data structures across the pipeline.

3. **Transparency**: The full evidence chain is preserved and exposed to users—sources, reliability ratings, stances, and reasoning are all traceable from final verdict back to original source.

4. **Progressive Enhancement**: The pipeline operates with graceful degradation (e.g., fallback to snippet if scraping fails, fallback to proxy if rate-limited) rather than failing entirely.

5. **Cost-Conscious Design**: Configurable limits on maximum claims, queries, and results prevent runaway API costs and control latency during development and production.

6. **Reproducibility**: Deterministic LLM outputs using zero temperature, structured YAML configurations, and comprehensive experiment tracking enable reproducible research.

7. **Separation of Concerns**: Classical algorithms handle tasks like reliability scoring, deduplication, and quality calculation, reserving LLM calls for tasks requiring reasoning and language understanding.

3.2 Technology Stack

3.2.1 Core Technologies

Table 1 presents the core technologies employed in the implementation.

Table 1: Core technology stack

Category	Technology	Version	Purpose
Language	Python	3.12	Core implementation
LLM Framework	Pydantic AI	$\geq$1.0.0	Agent orchestration, structured outputs
Data Validation	Pydantic	$\geq$2.0.0	Schema definitions, runtime validation
Web Framework	FastAPI	$\geq$0.115.0	REST API with SSE streaming
HTTP Server	Uvicorn	$\geq$0.32.0	ASGI server
Async HTTP	httpx	$\geq$0.28.1	Async HTTP client
Web Scraping	Selenium	$\geq$4.15.2	JavaScript-rendered content extraction
YouTube	youtube-transcript-api	$\geq$1.2.2	Transcript extraction
Domain Info	python-whois	$\geq$0.8.0	Domain age lookup
CLI	Typer	$\geq$0.15.0	Experiment runner CLI
Analysis	pandas, matplotlib	-	Data analysis and visualization

3.2.2 Large Language Models

The system supports multiple LLM providers to enable comparison of cost-quality trade-offs. Table 2 shows the available models and their configurations.

Table 2: LLM providers and pricing

Provider	Model	Context	Pricing (per 1M tokens)	Use Case
OpenAI	gpt-4o-mini	128K	\$0.15 / \$0.60	Default
OpenAI	gpt-4o	128K	\$5.00 / \$15.00	High-quality
OpenAI	gpt-4-turbo	128K	\$10.00 / \$30.00	Legacy
Ollama	qwen3:8b	40K	Free (local)	Budget/offline
Ollama	qwen3:4b	256K	Free (local)	Budget/offline

Large language models such as GPT-4 demonstrate strong performance across diverse reasoning benchmarks [29]. Despite these advances, models still suffer from hallucinations and are constrained by limited context windows, underscoring the need for careful configuration and reliability safeguards [29]. The model management layer makes it straightforward to switch providers, tune limits, and keep outputs deterministic through shared configuration. The architecture is extensible, allowing integration of additional LLM providers (such as Anthropic Claude, Google Gemini, or other OpenAI-compatible APIs) and local models via Ollama with minimal configuration changes.

The same layer also exposes a zero-cost, offline alternative through Ollama with Qwen 3 variants (qwen3:8b, qwen3:4b). Switching between OpenAI and local providers requires only a configuration change, enabling experiments that balance latency, quality, and hardware constraints. GPT-4o-mini remains the default selection for this thesis because cloud inference provides lower latency than consumer hardware while maintaining strong reasoning quality.

3.2.3 External Services

The system integrates with external services for search and transcript extraction:

- **YouTube Transcript API**: Primary transcript extraction service accessed via the youtube-transcript-api library, providing timestamped transcript segments.
- **Serper API**: Google Search wrapper providing organic search results with approximately 2,500 queries per month on the free tier.
- **YouTube oEmbed API**: Video metadata retrieval without authentication.

- **Webshare Proxy**: Rate limit bypass for transcript extraction with configurable proxy locations.

3.3 Pipeline Components

3.3.1 Transcriptor

The Transcriptor component extracts YouTube video transcripts with precise timestamp information for later claim-to-video mapping.

Implementation Details The transcriptor uses the youtube-transcript-api library to fetch available transcripts, with preference for English. When rate-limited by YouTube, it automatically falls back to a proxy service (Webshare). Key features include:

- **Timestamped Segments**: Each segment preserves `start` time and `duration` in seconds.
- **Character Position Mapping**: Enables mapping claim text positions back to video timestamps.
- **Title Fetching**: Uses YouTube oEmbed API to retrieve video title for context.
- **Proxy Fallback**: Automatic retry through Webshare proxy when rate-limited.

Output and Timestamp Mapping Each run returns the video identifier, the full transcript text, and a list of timestamped segments containing text, start time, and duration. A lightweight mapping pass scans the segments sequentially while maintaining a cumulative character counter so that claim positions can be translated into timestamps. This enables the interface to jump straight to the moment where a claim appeared without duplicating transcript parsing logic elsewhere.

3.3.2 Claim Extractor

The Claim Extractor identifies factual, verifiable claims from video transcripts using LLM-based extraction with thesis-relative importance ranking. This approach builds on prior work in automated claim detection: supervised models trained on annotated political debates have been used to detect check-worthy claims [11], and end-to-end systems like ClaimBuster monitor public discourse and prioritize factual statements for manual fact-checking [12]. These systems show that focusing on salient, verifiable claims improves the efficiency of fact-checking pipelines.

LLM Configuration The claim extractor uses deterministic settings for reproducibility: temperature set to 0.0 to ensure consistent outputs across multiple runs, max tokens limited to 1,200 to control response length and latency, and automatic retry logic (3 attempts) to handle transient LLM failures gracefully.

Prompt Engineering Strategy The claim extractor employs a novel approach with multi-step reasoning designed to prioritize claims most critical to the video's central argument:

Step 1: Thesis Inference — Before listing claims, the LLM infers the video's central thesis in no more than 25 words (e.g., "Climate change alarmism is driven more by politics and media than by settled science").

Step 2: Importance Ranking with Thesis Impact Test — Claims are scored based on their impact on the video's thesis using the question: "If this claim were proven false, would the thesis collapse or materially weaken?" Table 3 presents the scoring guidelines.

Table 3: Claim importance scoring guidelines

Score Range	Description	Examples
0.85–1.0	Prescriptive/causal claims undermining thesis	Policy proposals, causal mechanisms
0.60–0.80	Quantitative/historical evidence tied to thesis	Statistics, dates, expert citations
0.30–0.55	Context/supporting background	Definitions, general facts
0.0–0.25	Peripheral/anecdotal details	Personal stories, credentials

Step 3: Relevance Guardrails — Pure credential facts are capped at 0.30 unless the thesis questions expertise; statements not affecting the thesis are capped at 0.25; pure opinions are excluded; and paraphrases and duplicate numbers are removed.

Prompt Engineering Techniques The system employs several established prompting techniques across its components:

- **Role/system prompting**: Each component receives a focused system prompt that spells out its role, constraints, and expected outputs for consistent behavior.
- **Structured outputs**: Responses are constrained to typed schemas so downstream stages always receive validated fields.
- **Chain-of-thought reasoning**: The multi-step approach requires the model to state the thesis and then rate claims against it, which improves prioritization.

- **Dynamic instruction injection**: Runtime parameters such as `max_claims` are inserted directly into prompts so component behavior adapts without duplicating templates.
- **Deterministic generation**: Temperature stays at 0.0 across the pipeline for reproducible, debuggable outputs.
- **Zero-shot prompting**: Components receive task instructions without example I/O pairs, minimizing prompt length and avoiding format bias.

Post-Processing: Claim Localization After LLM extraction, each claim is located in the original transcript using fuzzy string matching. The algorithm applies a sliding window approach to find the best matching region, yielding character-level positions that enable precise timestamp mapping back to the video.

3.3.3 Query Generator

The Query Generator produces diverse, prioritized search queries tailored to each claim. A single LLM call emits direct restatements, synonym-heavy variations, source-focused prompts, and broader context queries together with priority tags so downstream steps can respect latency budgets. Guardrails ensure the suggestions include relevant entities, time periods, and qualifiers while capping each claim at a manageable number of high-yield searches.

Query Type Taxonomy The system generates four types of queries with different search strategies, as shown in Table 4.

Table 4: Query type taxonomy

Type	Description	Strategy	Example
DIRECT	Exact claim phrasing	Verbatim search	"unemployment rose 15% Q3 2024"
ALTERNATIVE	Rephrased with synonyms	Semantic variation	"jobless rate increase third quarter"
SOURCE	Target authoritative sources	Source-seeking	"BLS unemployment statistics Q3"
CONTEXT	Broader context	Background search	"economic indicators fall 2024"

Priority System Queries are prioritized 1–5 based on likelihood of finding reliable, definitive information: priority 1 queries are always included, priority 2 by default, priority 3 if budget allows, and priorities 4–5 rarely or only for completeness.

Context-Aware Query Generation Beyond factual accuracy, the query generator is designed to detect misleadingly framed claims—statements that may be technically accurate but presented without essential context. The system prompt instructs the LLM to respect temporal context when choosing keywords (e.g., including relevant years or qualifiers to avoid mixing eras), and to explicitly generate queries seeking counter-arguments or opposing views. This approach helps surface evidence that may qualify, limit, or contextualize the original claim, enabling more nuanced verdict generation.

3.3.4 Online Search

The Online Search component implements a multi-step pipeline to retrieve, assess, and extract evidence from web sources with adaptive quality filtering. Unlike the other LLM-based components, Online Search orchestrates multiple classical algorithms alongside a single LLM call for evidence extraction. This hybrid approach balances speed, reliability, and reasoning capabilities.

The reliability assessment combines domain-level heuristics with the Media Bias/Fact Check (MBFC) methodology, which employs a comprehensive weighted scoring system to evaluate media outlets' ideological bias and factual reliability [22]. These scores, combined with stance-aware evidence extraction instructions, keep unreliable or speculative passages from flowing downstream.

Figure 2 illustrates the four-step Online Search pipeline executed for each query.

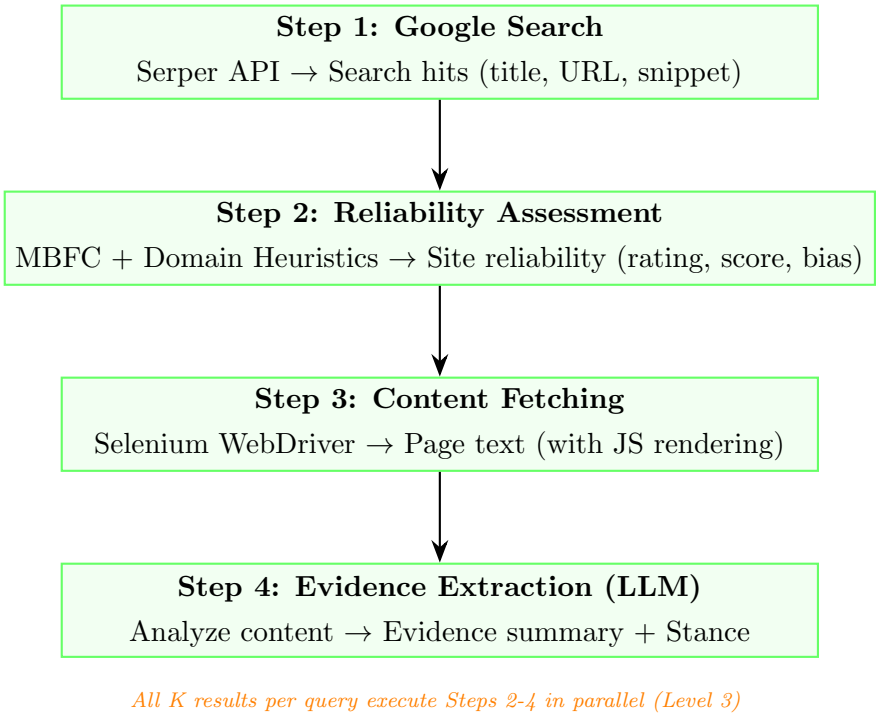


Figure 2: Online Search pipeline showing the four sequential steps executed for each search result. Steps 2–4 run in parallel across all K results per query (Level 3 parallelization).

Step 1: Google Search (Serper API) The Google search client wraps the Serper API for asynchronous search execution. The implementation uses persistent HTTP connections for performance and returns structured search hits containing title, URL, and snippet fields. Each query can retrieve up to 10 results, with the limit parameter controlling the exact number returned. The async design enables parallel query execution across multiple claims simultaneously.

Step 2: Website Reliability Assessment The reliability checker uses a multi-factor scoring system with first-match priority, combining external datasets with algorithmic heuristics:

- **Media Bias/Fact Check (MBFC) Dataset**: The system loads timestamped JSON snapshots containing nearly 10,000 news sources with credibility ratings (dataset extracted December 2025). Credibility levels are mapped to numerical scores: high (0.85), medium (0.60), low (0.30), and very low (0.15).
- **TLD Reputation**: High-trust top-level domains (.gov, .edu, .int) receive a base score of 0.90, reflecting their institutional authority.
- **Domain Age via WHOIS**: Domains $\geq$10 years old receive a +0.10 bonus (established

presence), while domains <1 year old receive a –0.15 penalty (recent creation may indicate lower trust).

The output includes a categorical rating (high, medium, low, unknown), numerical score (0.0–1.0), reasoning for the assessment, and political bias classification when available from MBFC data.

Step 3: Content Fetching (Selenium) Content fetching uses headless Chrome with smart waits: the scraper first grabs paragraph elements immediately, then allows extra rendering time only when less than 100 characters were captured. Images are disabled, page-load and wait timeouts cap the work (20 seconds and 12 seconds, respectively), and the cleaned text is trimmed to 8,000 characters before being passed to the LLM. Blocking Selenium calls are delegated to background threads so multiple results can be processed in parallel without freezing the async loop.

Step 4: Evidence Extraction (LLM) The evidence extractor analyzes retrieved content against claims using structured stance definitions:

- **SUPPORTS**: Evidence confirms or validates the claim through direct statements, semantic equivalents, or mechanism descriptions.
- **REFUTES**: Evidence contradicts or disproves the claim through counter-evidence or statements that evidence is unproven/disproven.
- **MIXED**: Both supporting and refuting elements present.
- **UNCLEAR**: Genuinely ambiguous content that discusses related topics without addressing the specific claim.

Critical prompt instructions ensure that mere discussion equals UNCLEAR, that mechanisms are recognized even without exact terminology, and that both Google snippets and page content are considered with better evidence prioritized. Importantly, the evidence extractor pays special attention to qualifiers such as “only”, “never”, “always”, and temporal scope limitations (e.g., “since X date”)—this enables detection of claims that may be technically accurate but misleadingly framed due to omitted context or overgeneralization.

Adaptive Credibility Filtering The search orchestrator implements adaptive credibility filtering as a key innovation for ensuring evidence quality. The algorithm operates in three phases:

1. **Initial Batch**: Fetch 2× the desired limit to provide filtering margin
2. **Quality Check**: If >50% of results are unreliable, fetch an additional batch to increase the pool of high-quality sources
3. **Intelligent Filtering**: Sort all results by reliability score and select the top reliable sources, with a minimum guarantee ensuring at least some results are returned even if reliability is universally low

Additional filtering mechanisms include stance filtering (removing unclear results if >50% have definitive stances) and URL deduplication using sets to prevent duplicate sources across different queries for the same claim.

3.3.5 Output Generator

The Output Generator synthesizes evidence into coherent verdicts with confidence levels, quality scoring, and timestamp mapping.

Two-Step Process The Output Generator employs a hybrid approach combining algorithmic evidence organization with LLM-based synthesis:

Step 1: Build Evidence Bundle (Algorithmic) — The system groups evidence by stance (supports, refutes, mixed, unclear), deduplicates sources by URL, and sorts within each group by reliability rating (high first), then numerical score, with alphabetic tie-breaking for consistency.

Step 2: Generate Verdict (LLM) — Organized evidence is formatted into a structured prompt containing stance labels, source counts, reliability ratings, and evidence summaries. The system prompt instructs the LLM to synthesize concise verdicts, naming sources explicitly only when clarifying contrasting perspectives or when evidence directly conflicts. This reduces verbosity while maintaining attribution transparency.

Evidence Quality Score The quality score is calculated algorithmically for consistency and speed, combining three weighted components: a base score for having any evidence, an actionable stance bonus scaled by the number of definitive sources, and a reliability bonus based on source credibility ratings. This score is used to rank claims in the user interface, prioritizing high-confidence verdicts.

Integrated Verdict Generation Verdicts are generated immediately after each claim's evidence collection completes, eliminating the latency overhead of waiting for all claims to finish.

After all parallel tasks complete, reports are sorted by quality score to prioritize high-confidence verdicts in the user interface.

3.4 Latency Optimization Strategies

The pipeline includes a set of latency optimizations following established principles for LLM application development [30]. These strategies enable scalable processing without compromising response quality.

3.4.1 Model Selection

The default model (gpt-4o-mini) is selected for its balanced performance across speed, cost-effectiveness, and large context window (128K tokens). This model provides sufficient reasoning capabilities for fact-checking tasks while maintaining low latency and competitive pricing ($0.15 per million input tokens, $0.60 per million output tokens). For budget-conscious deployments or offline operation, local Ollama models (qwen3:8b, qwen3:4b) offer zero-cost inference at the expense of potential quality degradation.

3.4.2 Output Constraints

Each component has carefully tuned token limits to minimize generation latency and API costs without sacrificing information quality. Claims are limited to approximately 40 words (sufficient for most factual assertions), context descriptions to 20 words (brief background), and evidence summaries to 1–2 sentences (key findings only). These constraints are enforced through explicit prompt instructions.

3.4.3 Content Trimming

Input token counts are minimized through aggressive content trimming strategies. Web content is trimmed to 6,000–8,000 characters before being passed to the Evidence Extractor, removing excessive context while retaining the most relevant portions (typically the first several paragraphs of an article). Evidence prompts include only Google snippets and extracted page text, explicitly excluding raw HTML, JavaScript, CSS, and other non-content elements that would inflate token counts without improving extraction quality.

3.4.4 Combined Operations

The pipeline minimizes LLM API calls by combining operations into single requests wherever possible. Each component makes exactly one LLM call per input unit (one call for claim

extraction, one per claim for query generation, one per search result for evidence extraction, and one per claim for verdict synthesis), with no multi-turn conversations that would multiply request counts.

3.4.5 Three-Level Async Architecture

The system implements three levels of nested parallelization to maximize throughput while maintaining dependency ordering:

- **Level 1 (Claims)**: After extracting N claims from the transcript, all claims are processed in parallel. Each claim independently proceeds through query generation, evidence search, and verdict generation. Critically, each claim's verdict is generated immediately after its evidence collection completes, rather than waiting for all claims to finish—this optimization reduces perceived latency by producing results progressively.
- **Level 2 (Queries per Claim)**: Within each claim's processing, the Query Generator produces M queries. These queries are executed in parallel, enabling simultaneous search across different query formulations (direct, alternative, source, context).
- **Level 3 (Search Results per Query)**: Within each query's execution, the Online Search component retrieves K results from Google. The four-step pipeline (reliability assessment, content fetching, and evidence extraction) runs in parallel for all K results, with each result processed independently.

This design achieves maximum theoretical parallelization of $N \times M \times K$ operations during the search phase, bounded only by system resources and API rate limits.

3.4.6 Real-Time Streaming

Server-Sent Events (SSE) provide progressive updates as the pipeline executes, improving perceived responsiveness for users. Each update labels the active stage (e.g., "claim extraction", "processing_claim_2", "generating_report") alongside a percentage so the UI can display the exact component currently running. Extracted claims are streamed immediately after the Claim Extractor finishes, enabling preview and early user feedback while later stages gather evidence.

3.4.7 Classical Methods for Non-Reasoning Tasks

Not every operation requires LLM reasoning. Deterministic tasks are handled by classical algorithms, which are faster, cheaper, and more reproducible. Table 5 shows these operations.

Table 5: Operations using classical methods

Operation	Method	Rationale
Reliability scoring	Rule-based + MBFC lookup	Faster, deterministic, no API cost
Claim localization	Fuzzy string matching	No LLM needed for text search
Evidence quality score	Algorithmic calculation	Consistent, fast, reproducible
URL deduplication	Hash set	O(1) lookup
Stance filtering	Threshold-based	Simple percentage check

3.5 Experimentation and Evaluation Framework

Evaluating LLM-based fact-checking systems presents unique challenges: outputs are non-deterministic, external dependencies (web search) introduce variability, and traditional benchmarks risk overfitting [33]. To address these challenges, a three-component experimentation framework was developed.

3.5.1 Experiment Runner

The experiment runner executes the fact-checking pipeline on batches of videos defined in a YAML configuration file. It invokes the tracker for each run and supports parallel execution across multiple videos.

3.5.2 Run Tracker

The tracker captures complete execution traces for reproducibility and analysis. Each run generates structured artifacts containing: run configuration, complete records of all LLM calls (prompts, responses, latency, cost), extracted claims and verdicts, and aggregated metrics. A decorator instruments the Pydantic AI agent methods to automatically record token counts and costs for every inference call.

3.5.3 Evaluator

The evaluator computes performance metrics across multiple dimensions:

- **Claim extraction**: Precision@k, Recall, F1, and MAP using semantic similarity matching against ground truth claims.
- **Verdict accuracy**: Comparison of system stances against ground truth labels.
- **Evidence retrieval**: Success rate and source reliability distribution.

- **System efficiency**: Latency and cost aggregation across pipeline components.
- **LLM-as-judge**: A language model assesses output quality on dimensions difficult to capture with traditional metrics [32].

Results include per-video reports and aggregate statistics across all evaluated videos.

3.6 API Layer and User Interface

3.6.1 FastAPI Setup

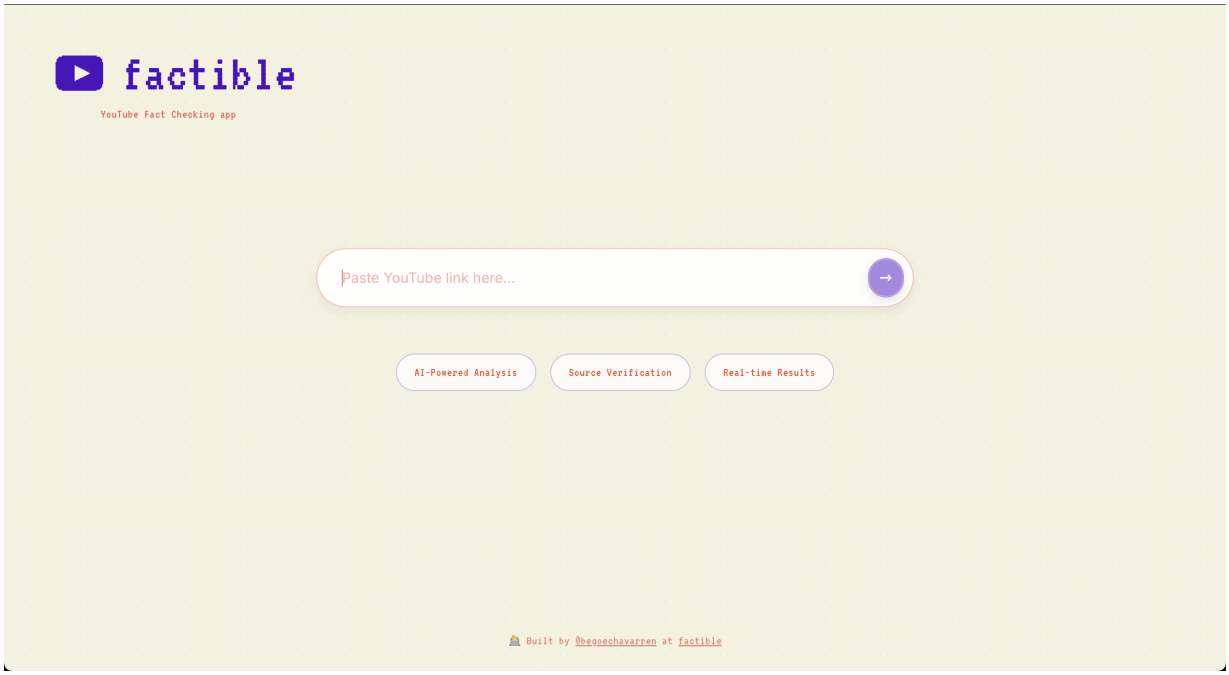
The API uses FastAPI with CORS middleware configured for local frontend development, supporting common development server ports.

3.6.2 Streaming Endpoint with SSE

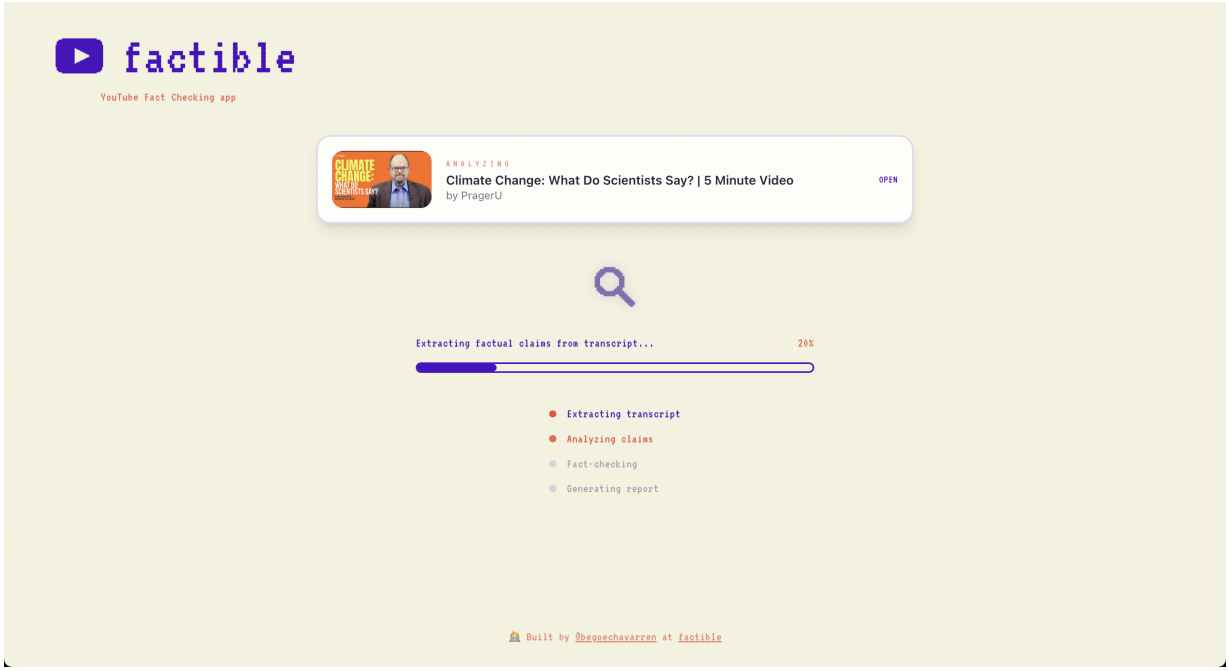
The streaming endpoint exposes the fact-checking run as a server-sent event stream. Progress callbacks enqueue updates that the API emits in real time, allowing clients to maintain a single open connection while receiving incremental status messages. Events follow a fixed sequence (transcript extraction, claim extraction, per-claim processing, verdict generation, completion) so the frontend can map each update to user-facing milestones without polling.

3.6.3 User Interface

A web-based frontend provides an accessible interface for end users to interact with the fact-checking pipeline. The interface is built with React and communicates with the backend via the streaming API endpoint. Figures 3 and 4 present the four main interface states. The interface emphasizes transparency by displaying source reliability ratings, confidence levels, and direct links to evidence sources.

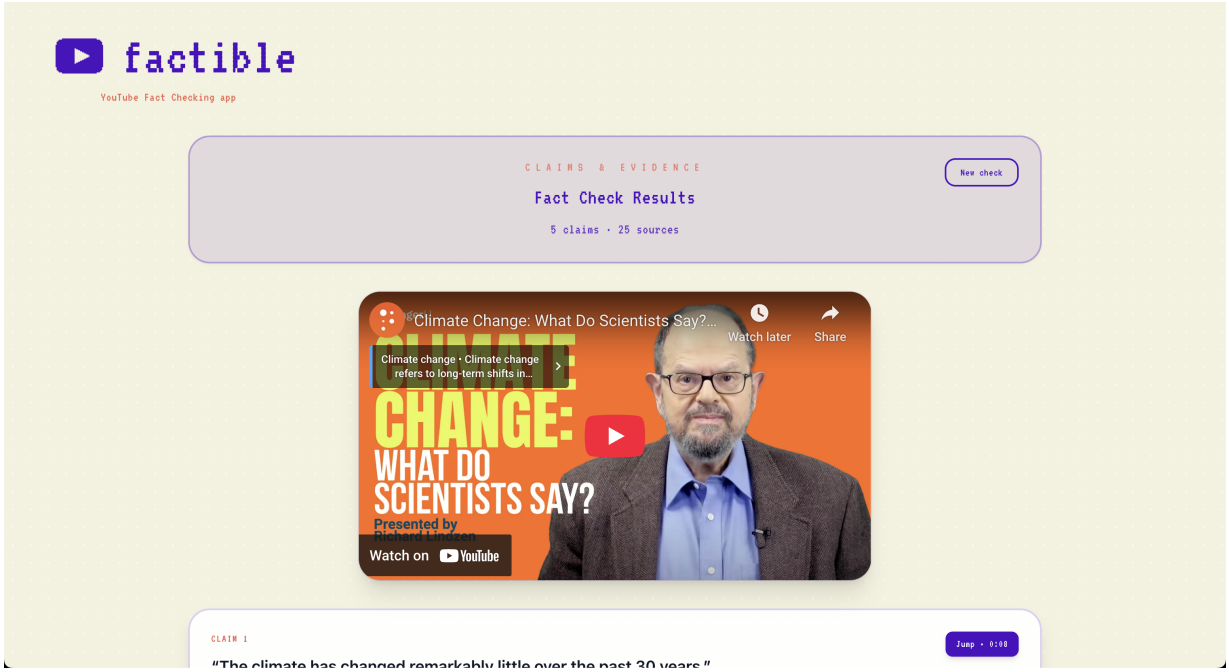


(a) Landing page with YouTube URL input

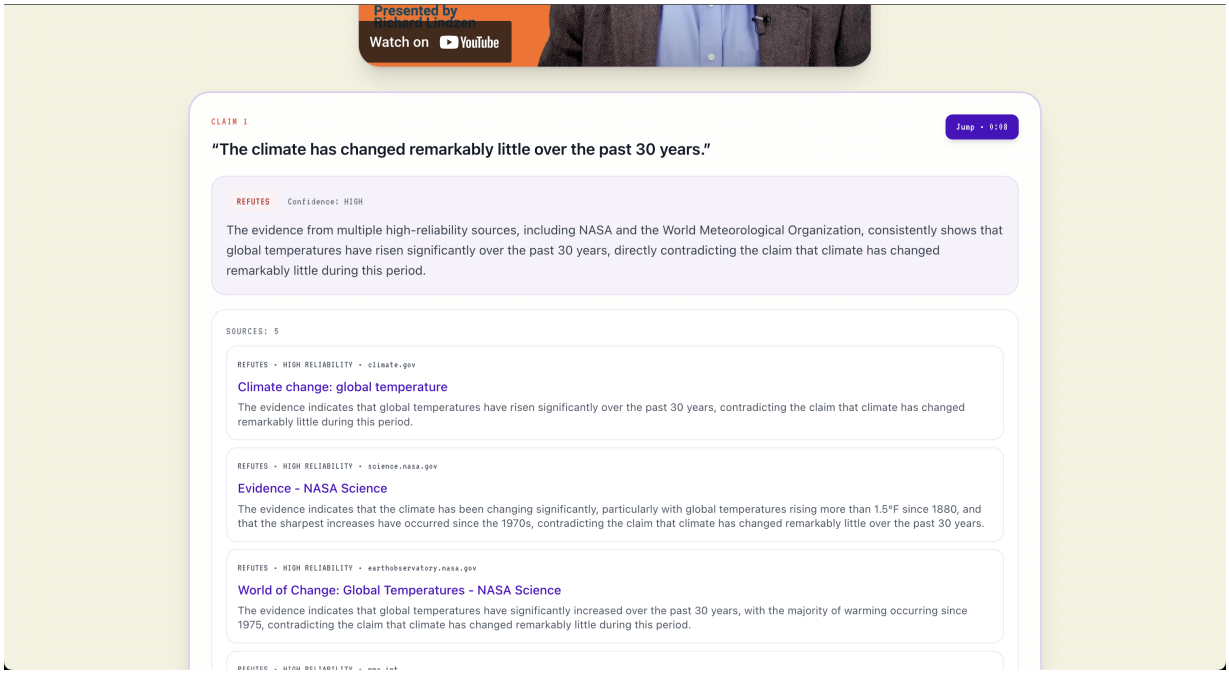


(b) Real-time processing with progress updates

Figure 3: User interface: input and processing stages.



(a) Results overview with embedded video player



(b) Detailed claim view with verdict and sources

Figure 4: User interface: results and claim detail views. Each claim shows its verdict, confidence level, and supporting evidence with source reliability ratings.

3.7 Engineering Practices

The thesis emphasizes data science outcomes, yet maintaining consistent engineering habits keeps experiments reproducible and easier to debug. Practices are kept lightweight and focused on what directly supports the fact-checking pipeline:

- **Typed schemas**: Pydantic models plus strict mypy checks prevent interface drift between components while supplying clear validation errors during runs.
- **Logging & fallbacks**: Module-level logging records claim/query context, and simple retry rules (e.g., fall back to Google snippets when scraping fails) keep long runs from collapsing on transient issues.
- **Centralized configuration**: Environment variables store secrets, YAML files define experiment batches, and a small Python settings module captures model limits, making it easy to reproduce or tweak runs.
- **Async + background threads**: Claims, queries, and search results fan out via asyncio gather calls, while Selenium/WHOIS work shifts to threads so blocking I/O never stalls the event loop; the same progress callbacks drive the SSE stream for user feedback.
- **Pre-commit hooks**: Lightweight automated checks run before every commit to keep formatting, linting, and type rules consistent.

4 Results

This section presents the experimental evaluation of Factible across 30 YouTube videos spanning diverse topics including health, science, politics, and climate. The evaluation framework follows established practices from claim detection and fact-checking research [11, 37], combining ground truth comparison with LLM-as-judge quality assessments. Every run was constrained to five claims per video, a setting chosen after analyzing the precision-recall tradeoff across multiple configurations while balancing cost and latency constraints.

4.1 Experimental Setup

4.1.1 Evaluation Dataset

The evaluation corpus consists of 30 YouTube videos manually annotated with ground truth claims. This sample size is comparable to evaluation scales used in end-to-end fact-checking systems; for example, ClaimBuster evaluated their system on 25 presidential debates [12].

Videos were selected across three thematic categories—climate, health, and political/social issues—with 10 videos per category to ensure balanced representation. Within each category, videos were equally split between factual content (5 videos) and misinformation (5 videos), allowing evaluation of the system's performance across different truth orientations. Videos range from educational science content to political commentary, representing diverse domains and claim densities. Table 6 summarizes the dataset characteristics.

Table 6: Evaluation dataset statistics

Metric	**Value**
Total videos	30
Ground truth claims per video (mean)	16.8
Ground truth claims per video (range)	9–35
Total ground truth claims	503
System claims extracted per video	5

4.1.2 Ground Truth Annotation

Ground truth annotations were created following a structured protocol. For each video, all factual, verifiable claims from the transcript were manually annotated. Each claim was annotated with an importance score (0.0–1.0) reflecting its centrality to the video's main argument, and a verdict label indicating the expected verification outcome (SUPPORTS, REFUTES, MIXED, or UNCLEAR). The annotation process followed guidelines from ClaimBuster's check-worthiness criteria [11], prioritizing claims that are specific, verifiable, and consequential. The evaluation dataset summary, including the list of evaluated videos, annotation guidelines, and aggregate statistics, is provided in Appendix A. The complete claim-level annotations are available in the project repository.

4.1.3 Evaluation Metrics

The evaluation employs metrics at two levels: claim extraction quality and verdict accuracy.

Claim Alignment via Semantic Similarity System claims are matched to ground truth using sentence-transformer embeddings (all-MiniLM-L6-v2). Claims whose cosine similarity exceeds 0.7 count as matches; others become false positives or false negatives. A greedy pass ensures each ground truth claim pairs with at most one system claim. This semantic alignment yields the true positives needed for precision, recall, F1, and MAP calculations.

Claim Extraction Metrics:

- **Precision@k**: Proportion of extracted claims matching any ground truth claim, using semantic similarity matching with a threshold of 0.7. This metric follows the standard information retrieval formulation used in claim detection systems [13].
- **Recall@k**: Proportion of ground truth claims matched by the top-k extracted claims [13].
- **F1 Score**: Harmonic mean of precision and recall.
- **Mean Average Precision (MAP)**: Ranking quality metric from information retrieval, measuring whether important claims appear early in the extraction order [13].
- **Recall@Important**: Recall specifically for high-importance claims (importance ≥ 0.80).
- **Importance-Weighted Coverage**: Percentage of total ground truth importance mass captured by matched claims.
- **Verdict Stance Accuracy**: Classification accuracy for the four-class stance problem (SUPPORTS, REFUTES, MIXED, UNCLEAR), measuring the percentage of verdicts matching ground truth stance.

System Efficiency Metrics:

- **Latency**: End-to-end processing time per video.
- **Evidence Retrieval Rate**: Proportion of queries successfully retrieving evidence.
- **Source Reliability**: Distribution of source reliability ratings across retrieved evidence.

4.1.4 Component Evaluation Scope

The evaluation focuses on components where the system makes reasoning decisions that can be compared against ground truth. Specifically, the evaluation covers **Claim Extraction** (precision, recall, importance ranking) and **Verdict Generation** (stance accuracy, explanation quality) as these components employ LLM-based reasoning that can produce varying results.

Components relying on external APIs—**Transcript Extraction** (YouTube Transcript API) and **Online Search** (Serper/Google)—are not evaluated directly, as their performance depends on third-party services rather than the system's design. However, their effectiveness is implicitly covered by the end-to-end evaluation: if transcript extraction fails, no claims can be extracted; if search fails, verdict accuracy degrades. The **Query Generator** component is assessed indirectly through evidence retrieval success rates, as effective queries should yield relevant evidence.

4.2 Precision-Recall Tradeoff Analysis

Before presenting detailed results, this section analyzes the precision-recall tradeoff to justify the choice of the configuration of maximum claims to fetch to be 5 (`max_claims=5`) as the primary configuration. The system was evaluated across six configurations with `max_claims` $\in \{1, 3, 5, 7, 10, 15\}$.

Table 7: Precision-recall tradeoff across different `max_claims` configurations

max_claims	Precision	Recall	F1	MAP
1	0.800	0.052	0.098	0.800
3	0.822	0.159	0.264	0.897
5	**0.813**	**0.262**	**0.390**	**0.870**
7	0.795	0.359	0.486	0.862
10	0.769	0.471	0.573	0.854
15	0.719	0.570	0.623	0.865

Figure 5 illustrates the precision-recall tradeoff. As `max_claims` increases, recall improves from 5.2% to 57.0%, while precision decreases from 82.2% to 71.9%. The slight precision increase from $k = 1$ (80.0%) to $k = 3$ (82.2%) indicates that the system benefits from extracting multiple high-confidence claims rather than being forced to select exactly one. Beyond $k = 5$, the classic precision-recall tradeoff becomes pronounced.

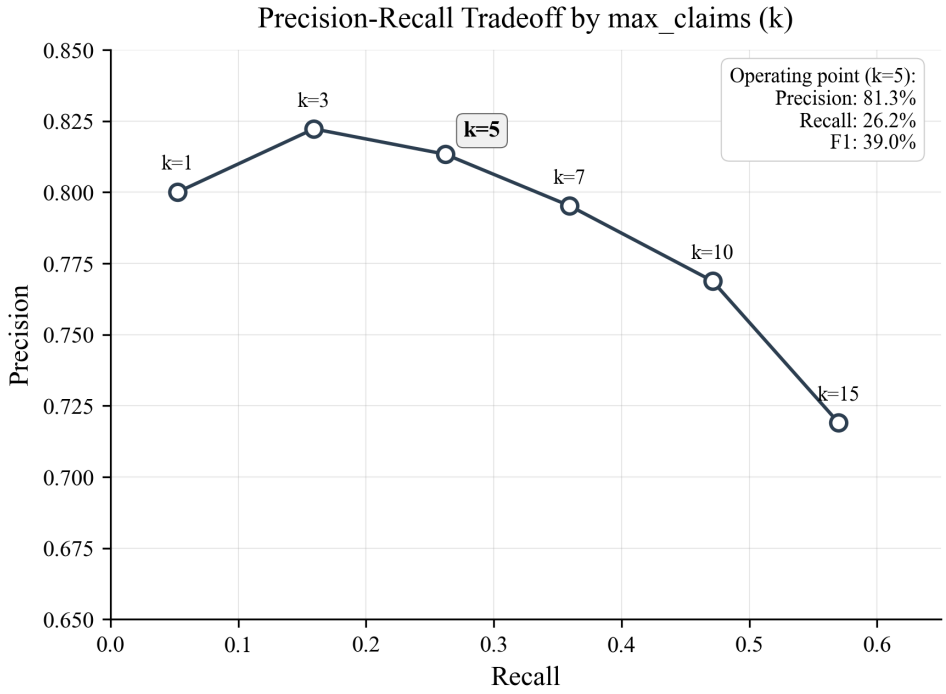


Figure 5: Precision-recall curve for different `max_claims` values. The selected operating point ($k = 5$) achieves 81.3% precision at 26.2% recall, balancing claim quality with coverage.

The configuration `max_claims=5` was selected as the primary operating point because it maintains high precision (81.3%) while achieving reasonable recall (26.2%), achieves a strong MAP score (0.870) indicating good ranking quality, and provides favorable cost and latency characteristics. Beyond `max_claims=5`, processing time and API costs increase linearly with claim count, while precision degrades. For a fact-checking application where user trust depends on accuracy and responsiveness, balancing precision, cost, and latency is critical—presenting 5 high-quality claims efficiently is more valuable than presenting 15 claims with higher false positive rates, increased costs, and longer wait times.

4.3 Claim Extraction Performance

Table 8 presents the claim extraction results at the primary configuration (`max_claims=5`).

Table 8: Claim extraction performance (n=30 videos, `max_claims`=5)

Metric	Mean	Std Dev
Precision@5	0.813	0.171
Recall	0.262	0.088
F1 Score	0.390	0.111
Mean Average Precision (MAP)	0.870	0.146
Recall@Important ($\geq$0.80)	0.395	0.178
Importance-Weighted Coverage	0.292	0.096
Importance MAE	0.126	0.060

4.3.1 Interpretation of Results

The system achieves 81.3% precision, meaning that approximately 4 out of 5 extracted claims on average match ground truth claims. This high precision indicates that the claim extractor successfully identifies legitimate factual claims rather than extracting irrelevant or fabricated statements.

The overall recall of 26.2% reflects the constraint of extracting only 5 claims from videos averaging 16.8 ground truth claims. However, the Recall@Important metric (39.5%) demonstrates that the system prioritizes high-importance claims—capturing nearly 40% of the most critical claims while only extracting approximately 30% of the total claim set. This selective extraction behavior aligns with the design goal of importance-based ranking, where claims are prioritized by their impact on the video's central argument.

The MAP score of 0.870 indicates strong ranking quality: important claims consistently appear early in the extraction order. This metric, inspired by ClaimBuster's evaluation framework [13], validates that the importance scoring mechanism effectively prioritizes claims.

The importance MAE of 0.126 (on a 0–1 scale) shows that system-assigned importance scores closely approximate human judgments, with typical errors of approximately one importance tier (e.g., scoring a claim 0.7 when ground truth is 0.85).

The 26.2% recall, while appearing low in isolation, must be interpreted in context. Given that videos contain an average of 16.8 checkable claims and the system extracts 5, a theoretical maximum recall of approximately 30% exists under this constraint. The achieved recall of 26.2% therefore represents strong performance relative to the configuration limit. Furthermore, the importance-weighted coverage of 29.2% indicates that the system captures nearly one-third of the total "importance mass" of ground truth claims. For practical fact-checking applications where user attention is limited, presenting 5 high-quality, important claims provides more value

than exhaustive but overwhelming coverage.

4.4 Verdict Generation Performance

Table 9 presents the verdict accuracy results.

Table 9: Verdict generation performance (n=30 videos)

Metric	Mean	Std Dev
Stance Accuracy	0.733	0.334

4.4.1 Accuracy Distribution Analysis

The standard deviation (0.334) in verdict accuracy is notable. Analysis of per-video results reveals a distribution skewed toward high accuracy:

- **21 videos (70.0%)** achieved high accuracy (≥75%)—the majority of extracted claims were correctly classified.
- **5 videos (16.7%)** achieved medium accuracy (25–74%)—partial verdict correctness.
- **4 videos (13.3%)** achieved low accuracy (<25%)—most claims were incorrectly classified.

The stance accuracy of 73.3% substantially exceeds a random baseline: for a four-class classification problem, random guessing would achieve approximately 25% accuracy, making this a 2.93× improvement. Moreover, unlike random classification, the system provides evidence-backed explanations that enable users to evaluate the verdict's reasoning.

With an evidence retrieval success rate of 94.7%, the system consistently finds relevant sources for most claims. Table 10 presents representative examples illustrating how verdict accuracy varies across the dataset.

Table 10: Verdict accuracy distribution with evidence retrieval rates (representative examples)

Video Topic	Retrieval Rate	Verdict Acc.	Avg Sources
High accuracy (≥75%) — 21 videos total, 4 examples shown			
Brain Benefits of Exercise (TED)	100%	100%	2.5
Climate Change (NatGeo)	100%	100%	2.3
Fossil Fuels	100%	100%	2.7
UK Election Results	100%	75%	2.8
Medium/Low accuracy (<75%) — 9 videos total, 4 examples shown			
Gender Wage Gap	100%	50%	2.7
FBI & January 6th	100%	50%	2.3
Inflation Explainer	60%	25%	1.8
A Nation of Immigrants	100%	0%	2.9

Notably, successful evidence retrieval does not guarantee high verdict accuracy. Some politically contentious topics achieve 100% retrieval but lower verdict accuracy, suggesting that the challenge lies not in finding evidence but in correctly synthesizing conflicting sources or matching the ground truth annotator's interpretation.

Analysis of low-accuracy videos reveals several contributing factors: (1) **contested claims**—topics with legitimate disagreement may have evidence supporting multiple stances, making definitive verdicts challenging; (2) **ground truth subjectivity**—some claims involve nuanced interpretations where reasonable annotators might disagree; and (3) **evidence-claim mismatch**—retrieved evidence may address related but not identical claims. These findings suggest that further improvements require enhanced evidence synthesis and more sophisticated handling of contested claims, rather than simply improving retrieval success.

4.5 Evidence Retrieval Performance

4.5.1 Retrieval Success

Table 11 summarizes evidence retrieval performance.

Table 11: Evidence retrieval performance (n=30 videos)

Metric	Value
Evidence retrieval success rate	94.7%
Average sources per query	1.52
Average evidence items per claim	2.41

The evidence retrieval success rate of 94.7% indicates that the vast majority of search queries return usable evidence. When evidence is retrieved, claims receive an average of 2.41 evidence items, providing multiple perspectives for verdict synthesis.

4.5.2 Source Reliability Distribution

A critical aspect of fact-checking is source quality. Table 12 presents the distribution of source reliability ratings across all retrieved evidence.

Table 12: Source reliability distribution (n=724 total sources)

Reliability Rating	Count	Percentage
High	605	83.6%
Medium	110	15.2%
Low	0	0.0%
Unknown	9	1.2%

The overwhelming majority of retrieved sources (83.6%) receive high reliability ratings from the Media Bias/Fact Check-based assessment system [22]. No sources received low reliability ratings, and only 1.2% were classified as unknown (typically due to missing MBFC data for niche domains). This distribution reflects both the reliability scoring heuristics and the retry behavior that fetches additional results whenever the initial batch skews toward low-reliability domains.

4.6 System Efficiency

4.6.1 Processing Latency

Table 13 presents processing time statistics.

Table 13: System latency (n=30 videos)

Metric	Value
Mean latency	129.6 seconds
Standard deviation	101.8 seconds
Minimum latency	36.5 seconds
Maximum latency	643.7 seconds
Total processing time (30 videos)	64.8 minutes

The mean processing time of 129.6 seconds per video enables near-interactive use for individual videos. The high variance (standard deviation 101.8s) reflects multiple contributing factors:

- **Video length**: Longer transcripts require more LLM tokens for claim extraction, increasing inference time.
- **Pipeline parameters**: The configuration parameters `max_claims`, `max_queries`, and `max_results_per_query` directly multiply the number of downstream operations. With the evaluation configuration (max_claims = 5, max_queries = 3, max_results = 3), each video triggers up to $5 \times 3 \times 3 = 45$ evidence extraction operations.
- **Evidence retrieval success**: Videos with failed evidence retrieval complete faster (fewer web requests), while videos requiring multiple successful searches experience longer latencies.
- **Web scraping variability**: JavaScript-heavy sites require longer Selenium wait times, and some domains respond slower than others.

4.6.2 Cost Analysis

All experiments used GPT-4o-mini for LLM inference at current pricing ($0.15 per million input tokens, $0.60 per million output tokens). Across the 30-video evaluation corpus, the average cost per video was $0.003, with individual videos ranging from $0.0008 to $0.0164 depending on transcript length, claim complexity, and evidence retrieval needs. The total cost for processing all 30 videos was $0.09, demonstrating the practical affordability of the system for individual users. This cost-effectiveness contrasts with concerns about LLM deployment costs noted in prior work [45], showing that fact-checking systems can achieve meaningful accuracy at minimal expense when using appropriately-sized models.

4.7 Qualitative Analysis

4.7.1 Successful Extraction Examples

Table 14 presents examples of successful claim extractions demonstrating semantic matching between ground truth and system-extracted claims.

Table 14: Examples of successful claim extraction with semantic matching

Ground Truth Claim	**System-Extracted Claim**	**Imp.**
A single workout immediately increases levels of neurotransmitters like dopamine, serotonin, and noradrenaline	A single workout increases levels of neurotransmitters like dopamine, serotonin, and noradrenaline	0.9
HIV infects one of the immune cells that is central to the body's response to pathogens—the helper T-cell	HIV infects helper T-cells, which are central to the immune response	0.95
Scientists at UCT have uncovered garlic's cancer fighting properties	Scientists at UCT uncovered garlic's cancer-fighting properties	0.95

These examples demonstrate that the system successfully extracts claims while allowing minor paraphrasing and condensation. The semantic similarity matching correctly identifies these as equivalent claims despite surface-level textual differences.

4.7.2 Error Analysis: Verdict Failures

Analysis of verdict errors reveals systematic patterns. With high evidence retrieval success (94.7%), the primary failure modes involve stance misclassification and handling nuanced claims where conflicting evidence requires domain expertise to synthesize correctly. Table 15 categorizes the primary error types.

Table 15: Verdict error categories

Error Type	Description
Evidence retrieval failure	No evidence retrieved; system defaults to UNCLEAR
Stance misclassification	Evidence retrieved but stance incorrectly assessed (e.g., MIXED classified as REFUTES)
Nuanced claims	Claims requiring domain expertise to evaluate mixed evidence

Error analysis reveals that evidence retrieval success does not guarantee verdict accuracy. Two of the four lowest-accuracy videos achieved 100% evidence retrieval but 0% verdict accuracy, indicating that the challenge lies in evidence synthesis rather than retrieval. These cases involve politically contested claims where ground truth stances require nuanced interpretation of conflicting sources.

4.7.3 Analysis of Low-Accuracy Cases

Only 4 videos (13.3%) achieved less than 25% verdict accuracy. Detailed analysis reveals their characteristics:

Table 16: Low-accuracy video analysis (<25% verdict accuracy)

Video Topic	Type	Retrieval	Accuracy	Likely Cause
Trump's Historic National Emergency	Factual	60%	0%	Contested political claims
A Nation of Immigrants	Misinfo	100%	0%	Conflicting sources
Juice vs. Whole Fruit	Factual	20%	20%	Limited evidence retrieval
Sleep & Teenage Brain	Factual	100%	20%	Nuanced scientific claims

Key observations:

- **High retrieval does not guarantee accuracy:** Two videos achieved 100% retrieval but 0–20% accuracy, confirming that evidence synthesis is the bottleneck for contested claims.

- **No single category dominates:** Low-accuracy videos span both political (2) and health (2) topics, and include both factual content (3) and misinformation (1).
- **Contested claims are hardest:** The common thread is claims where reasonable sources disagree or where ground truth requires nuanced interpretation.

Successful categories: Videos on scientific explanations (e.g., climate science, exercise benefits), health misinformation debunking (e.g., fluoride claims, detox myths), and conspiracy content (e.g., chemtrails, geoengineering) achieved high accuracy, demonstrating the system's effectiveness when evidence clearly supports or refutes claims.

4.8 Considerations for Generative AI Systems

It is important to contextualize these results within the unique characteristics of generative AI systems. Unlike traditional machine learning models with deterministic outputs, LLM-based systems introduce inherent variability that affects evaluation interpretation.

4.8.1 Non-Determinism in LLM Systems

Despite configuring all LLM calls with `temperature=0.0` to minimize output variability, complete determinism is not guaranteed. Even with zero temperature, LLM outputs may vary across runs due to:

- **Floating-point precision**: GPU computation introduces subtle numerical variations that can cascade through token selection.
- **Model updates**: Cloud-hosted models (e.g., GPT-4o-mini) may be silently updated by providers, affecting outputs over time.
- **Batching effects**: Different batch sizes or concurrent requests may influence internal state.

This inherent non-determinism means that exact reproduction of results is challenging, though setting temperature to zero substantially reduces variability compared to default settings.

4.8.2 External Dependencies and Temporal Sensitivity

Beyond LLM variability, the system's reliance on external web search introduces additional sources of result variability:

- **Search result volatility**: Web search results change over time as new content is indexed and rankings evolve.
- **Content availability**: Websites may become unavailable, paywalled, or block automated access.
- **Rate limiting**: Search APIs may throttle requests, causing some queries to fail during high-load evaluation runs.

These factors contribute to verdict accuracy variance: search results may differ between runs, and a claim that retrieved limited evidence in one execution might find more sources in another.

4.8.3 Implications for Metric Interpretation

Unlike traditional classification tasks where metrics are stable given fixed test data, LLM-based systems produce metrics with inherent variance. When evaluating generative AI systems, researchers should consider:

- **Expected variability**: Metrics may vary by several percentage points across identical evaluation runs.
- **Qualitative validation**: Beyond aggregate metrics, examining individual outputs provides crucial insight into system behavior.
- **Temporal context**: Results reflect system performance at a specific point in time with then-current search results and model versions.

The strategies employed in this work to maximize reproducibility—temperature=0.0 for all LLM calls, fixed random seeds, comprehensive logging, and experiment versioning—represent current best practices for LLM evaluation but cannot eliminate all sources of variability.

4.9 Comparison with Related Fact-Checking Systems

While direct performance comparison across fact-checking systems is constrained by differences in evaluation datasets, task definitions, and domain characteristics, contextualizing these results against established benchmarks and recent approaches provides valuable perspective on the system's contributions.

4.9.1 Foundational Benchmarks

The FEVER dataset [37] established the foundational benchmark for fact verification against textual sources, comprising 185,445 claims verified against Wikipedia with sentence-level evidence annotations. Their best baseline system achieved 31.87% accuracy when requiring correct evidence retrieval, using a pipeline of TF-IDF document retrieval, sentence selection, and decomposable attention for textual entailment. Notably, 16.82% of FEVER claims require multi-sentence reasoning, highlighting the complexity of evidence aggregation even in controlled environments. State-of-the-art systems on the FEVER leaderboard have since achieved 70–80% accuracy, demonstrating significant progress on this benchmark.

Building on this foundation, SciFact [40] introduced scientific claim verification with 1,409 expert-written claims against biomedical abstracts. Their VERISCI baseline achieved 46.5% F1 on the open retrieval task (AbstractLabel+Rationale), demonstrating that domain-specific adaptation—combining FEVER pretraining with in-domain fine-tuning—substantially improves performance over zero-shot approaches (36.4% F1). Their error analysis identified five key reasoning challenges: scientific background knowledge, directionality, numerical reasoning, cause-and-effect relationships, and coreference resolution—challenges that parallel the difficulties Factible encounters with YouTube content.

4.9.2 Post-hoc Verification and Attribution Approaches

The RARR framework [8] introduced iterative research-and-revision for improving attribution in LLM outputs. RARR employs a two-stage pipeline: a *research stage* that generates queries and retrieves evidence from web sources, and a *revision stage* that uses agreement models to detect disagreements between claims and evidence before making targeted edits. On diverse QA datasets including Natural Questions, StrategyQA, and QReCC, RARR achieved F1 scores (combining attribution and preservation) ranging from 43% to 68%.

A key insight from RARR relevant to this work is the tension between attribution improvement and content preservation. RARR found that editing to improve attribution while changing only 10–20% of text represents a careful balance—aggressive editing sacrifices the original intent while minimal editing leaves unattributed claims intact. Factible faces an analogous trade-off: the system prioritizes direct verification verdicts (Supported, Refuted, Insufficient Evidence, Unverifiable) over claim revision, trading the flexibility of correction for clearer actionable outputs suited to users assessing video content truthfulness.

VeriScore [36] and FIRE [43] extended this direction by introducing confidence-based retrieval triggers that reduce computational costs without sacrificing accuracy. FIRE's iterative retrieval-and-verification approach, which triggers external retrieval only when model confi-

dence falls below a threshold, parallels Factible's confidence-based handling of borderline claims.

4.9.3 Multi-Agent Architectures

Recent work has demonstrated the effectiveness of multi-agent LLM architectures for fact verification. FactAgent [45] employs a four-agent pipeline comprising Input Ingestion, Query Generation, Evidence Retrieval, and Verdict Prediction—an architecture closely resembling Factible's. Evaluated on HoVER, FEVEROUS, and SciFact-Open benchmarks with GPT-4o-mini, FactAgent achieved F1 scores of 0.600–0.617 on multi-hop reasoning tasks (HoVER 2–3 hop), 0.548–0.681 on FEVEROUS structured data tasks, and 0.770 on scientific claim verification (SciFact-Open), representing a 12.3% relative improvement over baseline methods. Notably, FactAgent integrates MBFC-based source credibility filtering—a design choice independently adopted in Factible—finding that credibility filtering significantly reduces usable links (to 9.6–15.8% on HoVER) but improves verdict quality by excluding unreliable sources. Their ablation studies showed that generating 3–4 queries per subclaim provides optimal balance between reasoning depth and evidence precision, aligning with Factible's configuration choices.

The MAD-Fact framework [27] provides particularly relevant methodological insights. MAD-Fact implements a three-tier multi-agent debate system comprising: (1) a *Clerk Agent* for atomic claim decomposition, (2) a *Jury* of Evaluator Agents with heterogeneous role assignments (Public, Critic, News Author, Scientist, etc.) that assess factuality through structured debate, and (3) a *Judge Agent* that aggregates verdicts via majority voting. On fact-checking benchmarks including FacToolQA and FELM-WK, MAD-Fact achieved F1 scores of 0.88 (Label=True) and 0.67 (Label=False), consistently outperforming single-model baselines.

Three findings from MAD-Fact are particularly instructive:

- **Debate dynamics**: Multi-agent systems exhibited human-like traits during debates—agents adhered to their viewpoints, actively corrected peers' mistakes, and engaged in self-reflection. This emergent behavior contributed to superior performance over single-agent systems.

- **Retrieval integration**: The "Autonomous Retrieval and Free Debate" rule, where agents decide based on confidence whether to invoke external retrieval, balanced knowledge utilization with computational efficiency.

- **Homogeneous vs. heterogeneous initialization**: Interestingly, initializing all agents with the same model (GPT-4o-mini) outperformed heterogeneous initialization with multiple model families, as cross-model agents exhibited higher frequencies of misleading each other and struggled to reach consensus.

Factible adopts a comparable multi-agent pipeline—comprising claim extraction, evidence retrieval, and verdict determination—but addresses a significantly more challenging domain: unstructured YouTube content with noisy automatic transcriptions and no gold evidence. Despite this increased difficulty, the 73.3% accuracy demonstrates that multi-agent approaches can generalize beyond curated datasets to real-world multimedia platforms.

4.9.4 Domain Difficulty Considerations

The primary challenge in cross-system comparison lies not in algorithmic sophistication but in domain difficulty. Most comparable systems operate on closed corpora with clean text, gold evidence, and pre-extracted claims. In contrast, Factible handles:

- **Noisy input**: Automatic transcription errors, missing punctuation, and speaker attribution ambiguity
- **Open-domain retrieval**: Web-scale evidence search without annotator-provided ground truth
- **Real-world claims**: Naturally occurring statements rather than synthetic mutations
- **Temporal dynamics**: Time-sensitive claims that may have been true at recording but become outdated

Table 17 summarizes key characteristics across systems. The performance gap between FEVER's baseline 31.87% on structured Wikipedia and Factible's 73.3% on unstructured YouTube does not indicate superior accuracy in absolute terms, but rather reflects fundamentally different difficulty profiles and evaluation methodologies. The system's ability to maintain reasonable accuracy while handling YouTube's compounding challenges represents meaningful progress toward bringing automated fact-checking to platforms where misinformation most commonly spreads.

Table 17: Comparison of fact-checking systems across key dimensions. Performance metrics are reported as originally published and are not directly comparable due to differing task definitions and evaluation protocols.

System	Corpus Type	Evidence	Claims	Metric	Performance
FEVER [37]	Wikipedia	Gold	Synthetic	Accuracy	31.87%
SciFact [40]	Scientific	Gold	Natural	F1	46.5%
RARR [8]	QA datasets	Retrieved	Natural	F1	43–68%
FactAgent [45]	Wikipedia	Retrieved	Natural	F1	51–77%
MAD-Fact [27]	Fact-check	Retrieved	Natural	F1	67–88%
Factible (ours)	YouTube/Web	Retrieved	Natural	Accuracy	73.3%

4.9.5 Implications for System Design

The comparative analysis yields several design implications validated by this implementation:

1. **Atomic decomposition is essential**: Following FactScore, SAFE, and MAD-Fact, decomposing claims into atomic, independently verifiable units enables fine-grained evaluation and targeted verdict assignment.

2. **Retrieval quality bounds verification quality**: As RARR demonstrated, even sophisticated reasoning cannot compensate for poor evidence retrieval. Factible's web search integration addresses this but introduces new challenges around source credibility assessment.

3. **Source credibility filtering is critical**: FactAgent's finding that MBFC-based credibility filtering reduces usable links to 9.6–15.8% on complex multi-hop tasks while improving verdict quality validates the independent adoption of this approach in Factible. The trade-off between evidence quantity and quality favors aggressive filtering.

4. **Multi-agent approaches mitigate single-model bias**: Single-model verification is vulnerable to systematic errors from hallucination or knowledge gaps. Factible's multi-stage pipeline, like FactAgent's four-agent architecture, distributes verification responsibility across specialized agents while maintaining computational efficiency.

5. **Domain adaptation requires careful evaluation**: Performance numbers from clean-text benchmarks do not transfer to noisy multimedia domains. Future work should develop standardized benchmarks for video fact-checking that account for automatic transcription errors and real-world claim distributions.

4.10 Summary of Key Findings

The experimental evaluation demonstrates that Factible achieves reliable fact-checking performance across diverse video content:

1. **High-precision claim extraction:** 81.3% precision with strong importance ranking (MAP 0.870), meaning users can trust that presented claims are legitimate, high-priority factual statements.

2. **Robust evidence retrieval:** 94.7% success rate with 83.6% of sources from high-reliability origins, demonstrating effective query generation and source filtering.

3. **Consistent verdict accuracy:** 73.3% overall accuracy, with 70% of videos (21/30) achieving $\geq$75% accuracy.

4. **Identified challenges:** The 4 low-accuracy videos (13.3%) involve politically contested claims with legitimate disagreement, where evidence synthesis rather than retrieval poses the challenge.

5. **Practical efficiency:** 129.6 seconds mean latency at \$0.003 per video enables cost-effective, near-interactive use.

These results position Factible as a reliable tool for preliminary fact-checking of YouTube content. The system performs well on scientific, health, and clearly verifiable claims, while contested political topics with conflicting evidence sources represent the primary remaining challenge for future work.

5 Conclusions

This section presents the conclusions of this thesis, evaluates the achievement of initial objectives, reflects on the methodology employed, and discusses ethical and sustainability considerations.

5.1 Contributions and Achievements

This thesis has presented Factible, a multi-agent system for automated fact-checking of YouTube videos. The work addresses critical gaps identified in the State of the Art—particularly the lack of end-to-end usability, dependence on structured sources, and the production readiness gap—demonstrating the practical viability of LLM-based verification systems for real-world deployment.

5.1.1 Technical Contributions

1. **End-to-End Video Fact-Checking Pipeline**: Unlike prior research that focuses on isolated components [12, 37], Factible implements a complete pipeline from YouTube URL to verified claims with evidence-backed verdicts. The system handles transcript extraction, claim detection, query generation, evidence retrieval, and verdict synthesis autonomously, addressing the end-to-end usability gap identified in Section 2 [19].

2. **Novel Claim Extraction Approach**: A novel prompting approach that prioritizes claims based on their impact on the video's central argument, achieving 81.3% precision and 0.870 MAP score for claim extraction.

3. **Open-Web Evidence Retrieval with Credibility Assessment**: Moving beyond Wikipedia and closed knowledge bases [1, 37], the system retrieves evidence from the live web using search engines, incorporating source credibility evaluation using Media Bias/Fact Check data and domain heuristics. This enables verification of claims about current events, specialized domains, and topics not well covered in encyclopedic sources.

4. **Three-Level Parallel Architecture**: An efficient execution model that parallelizes processing across claims, queries, and search results, achieving mean processing time of 129.6 seconds per video while maintaining cost under $0.01 per video. This addresses the performance and scalability limitations noted in the literature [19].

5. **Modular Multi-Agent Architecture**: Five specialized agents with structured Pydantic schemas enable transparency, maintainability, and independent optimization. This design facilitates experimentation with different models and techniques, demonstrating the practical benefits of multi-agent approaches [3, 45].

6. **Production-Ready Implementation**: A web-based interface with real-time SSE streaming makes the verification process transparent and accessible to non-expert users, bridging the gap between research prototypes and usable products [19]. The interface displays complete evidence chains including stance classifications, source reliability ratings, political bias indicators when available, evidence summaries, and confidence levels—enabling users to evaluate the system's reasoning rather than accepting verdicts blindly.

5.1.2 Research Contributions

1. **Annotated Evaluation Dataset**: A corpus of 30 YouTube videos with 503 manually annotated ground truth claims across health, climate, and political topics, providing a foundation for future research on video fact-checking.

2. **Comprehensive Evaluation Framework**: A methodology combining quantitative metrics (precision, recall, MAP), semantic similarity matching, verdict accuracy assessment, and system efficiency measurements. This multifaceted approach addresses the evaluation challenges discussed in the literature and provides a realistic assessment of system capabilities [32, 33].

3. **Analysis of Failure Modes**: Systematic identification of where automated fact-checking struggles, particularly with contested political claims where evidence synthesis rather than retrieval poses the challenge.

4. **Focus on Video Platform**: Unlike most fact-checking research that focuses on text-based content [12, 37], this work specifically addresses YouTube video verification, handling the complete video-to-verification workflow. This fills a significant gap where video platforms have been largely overlooked despite their scale and influence on information consumption.

5.1.3 Fulfillment of Objectives

The main objective—to design, implement, and evaluate a multi-agent system capable of automatically verifying factual claims in YouTube videos—was **fully achieved**. Evaluation on 30 videos demonstrates 81.3% claim extraction precision, 94.7% evidence retrieval success, and 73.3% verdict accuracy.

All six specific objectives defined in Section 1.2.2 were achieved. The technical and research contributions above address objectives 1–2 (architecture and implementation), 4 (evaluation), and 5 (end-to-end system). Two additional objectives merit explicit mention:

- **LLM usage optimization** (Objective 3): The system uses GPT-4o-mini with deterministic settings and token budgets. Cost, latency, and quality trade-offs were analyzed, achieving $0.003 average cost per video with 129.6 seconds mean latency. Prompt optimization strategies with structured outputs were implemented across all pipeline components.

- **Ethical and bias considerations** (Objective 6): The system incorporates source transparency, political bias indicators, multi-perspective evidence presentation, and clear confidence levels. Section 5.2 provides detailed discussion of ethical considerations and bias mitigation strategies.

5.2 Reflection on Ethical and Sustainability Considerations

5.2.1 Sustainability Assessment

The project's environmental impact was considered throughout development:

- **Computational efficiency**: Using GPT-4o-mini rather than larger models (GPT-4, GPT-4-turbo) significantly reduces energy consumption while maintaining adequate quality. The average cost of $0.003 per video reflects both financial and environmental efficiency.
- **Optimization strategies**: Token budgets, content trimming, and classical algorithms for non-reasoning tasks minimize unnecessary LLM calls.
- **Local model support**: The system supports Ollama-based local models, enabling zero-cloud-cost operation for users with appropriate hardware, further reducing the system's carbon footprint for some deployments.

While LLM usage inherently involves energy consumption, the system's design choices represent a conscious effort to balance capability with environmental responsibility.

5.2.2 Ethical Considerations

The system addresses ethical concerns through several mechanisms:

- **Transparency**: All evidence sources are exposed with URLs, reliability ratings, political bias indicators (when available from Media Bias/Fact Check data), and stance classifications, enabling users to verify the system's reasoning and identify potential source perspectives.
- **Multi-perspective presentation**: Evidence is organized by stance (supports, refutes, mixed, unclear), avoiding false certainty and acknowledging when claims are genuinely contested.
- **Confidence levels**: Verdicts include confidence ratings (low, medium, high), helping users calibrate trust appropriately.
- **Tool positioning**: The system is positioned as a fact-checking assistant rather than an authoritative source, emphasizing its role in supporting rather than replacing human judgment.

5.2.3 Potential Risks and Mitigation

Several risks warrant ongoing attention:

- **Over-reliance on automation**: Users might accept system verdicts uncritically. Mitigation: Clear messaging that the system provides preliminary assessment, not definitive truth.
- **Bias amplification**: LLMs may perpetuate biases from training data. Mitigation: Source diversity in evidence retrieval and transparent presentation of multiple perspectives.
- **Weaponization concerns**: The system could theoretically be misused to generate plausible-sounding refutations of true claims. Mitigation: Open-source release enables scrutiny, and the evidence-based approach makes manipulation detectable.

5.2.4 Sustainable Development Goals

As outlined in Section 1.3.2, the project contributes to SDG 4 (Quality Education) by serving as a media literacy tool that promotes critical thinking, SDG 16 (Peace, Justice and Strong Institutions) by supporting informed public discourse through transparent verification mechanisms, and SDG 10 (Reduced Inequalities) by democratizing access to fact-checking capabilities through its open-source, low-cost design.

5.3 Concluding Remarks

This thesis addresses a critical gap in automated fact-checking research: the lack of end-to-end verification systems for video platforms. It demonstrates that automated fact-checking of YouTube videos is not only feasible but can achieve practically useful performance levels. Factible represents a step toward democratizing access to fact-checking capabilities, providing users with tools to critically evaluate video content that might otherwise go unexamined.

The system's strengths—high precision claim extraction, robust evidence retrieval, transparent verdict presentation, and practical efficiency—make it suitable for preliminary fact-checking of YouTube content. Its limitations—particularly with contested political claims—highlight areas where human judgment remains essential and automated systems should complement rather than replace expert fact-checkers. Section 6 outlines directions for addressing these limitations and extending the system's capabilities.

As misinformation continues to evolve in volume and sophistication, automated fact-checking systems will become increasingly important. The modular, open-source nature of Factible provides a foundation for continued research and development in this critical domain. By making

both the system and its evaluation transparent, this work aims to advance the broader goal of building more informed and resilient information ecosystems.

The complete source code, evaluation dataset, and documentation are publicly available at https://github.com/begoechavarren/factible, enabling researchers and practitioners to build upon this work toward more informed and resilient information ecosystems.

6 Future Work

While the current implementation demonstrates the viability of automated fact-checking for YouTube videos, several limitations of the current approach motivate directions for future research. This section outlines these limitations alongside the corresponding opportunities for enhancement.

6.1 Multilingual Support

The system currently supports only English content, yet misinformation is a global phenomenon. Extending to languages such as Spanish, Portuguese, Hindi, or Arabic would require adapting prompts, identifying reliable sources in each language, and evaluating source credibility across different cultural and media contexts.

6.2 Large Language Model Comparison

Future studies should benchmark multiple commercial and local models (GPT-4 variants, Claude, Gemini, LLaMA, Qwen, Mistral) across pipeline components. Such comparisons would map cost-quality trade-offs and assess whether mixed deployments—where different models handle different stages—improve overall performance.

6.3 Enhanced Prompt Engineering

Maintaining a versioned prompt library with documented variants would make it easier to evolve instructions without rewriting code. Automated prompt tuning using evaluation feedback and domain-specific templates (scientific vs. political claims) could further improve consistency.

6.4 Enhanced Source Reliability Assessment

Source reliability could be refined by incorporating publisher-level signals such as impact factors and retraction history for academic sources, topic-aware weighting that distinguishes science

journalism from political commentary, and time-varying trust scores that account for reputation changes. Lightweight cross-referencing when conflicting evidence appears would further strengthen verdict quality.

6.5 Handling Contested Claims

The current system struggles with genuinely contested claims where reasonable sources disagree. Future approaches might explicitly detect and flag such claims as "contested" rather than attempting definitive verdicts, implement debate-style verification using multiple LLM agents with different perspectives, or develop specialized pipelines for political versus scientific claims. Incorporating user feedback could further refine handling of ambiguous cases.

6.6 Production Deployment

Transitioning from research prototype to production would involve cloud deployment with auto-scaling for handling concurrent users, user authentication and rate limiting to prevent abuse, and caching for transcripts and reliability assessments to reduce latency and costs. Monitoring infrastructure would ensure operational health, while a browser extension could enable seamless YouTube verification without leaving the platform.

6.7 Extended Evaluation

The current evaluation has constraints (single annotator, 30-video corpus, point-in-time results) that future work should address. A larger corpus of 100+ videos with multiple annotators would establish inter-annotator agreement baselines and enable more robust statistical conclusions. User studies could evaluate perceived usefulness and trust calibration, while longitudinal evaluation would track performance as LLMs and web content evolve over time.

6.8 Multimodal Verification

The current system operates only on transcript text, ignoring visual content. Future work could incorporate image and video frame analysis for visual claim verification, chart and graph interpretation for data-driven claims, speaker identification for credibility signals, and deepfake detection for authenticity verification.

7 Glossary

This section defines the key terms used throughout this thesis.

7.1 Terms

Automated Fact-Checking The use of computational methods to verify the accuracy of factual claims, typically involving claim detection, evidence retrieval, and verdict generation.

Chain-of-Thought (CoT) A prompting technique where a language model is instructed to explain its reasoning step by step before providing a final answer, improving accuracy on complex tasks.

Check-Worthiness A property of claims indicating they are both verifiable (can be confirmed or refuted with evidence) and important enough to warrant fact-checking effort.

Claim Detection The task of identifying factual statements within text that can be verified against external evidence, also known as claim extraction.

Evidence Retrieval The process of finding documents, passages, or data from external sources that support or refute a given claim.

Fact-Checking The process of verifying the accuracy of claims by examining available evidence, traditionally performed by journalists and specialized organizations.

Hallucination A phenomenon where language models generate plausible-sounding but factually incorrect or fabricated information.

Information Retrieval (IR) The field concerned with finding relevant documents or passages from large collections in response to a query.

LLM-as-Judge An evaluation paradigm where a language model assesses the quality of outputs from another model, providing scores or judgments on dimensions like coherence, factuality, or relevance.

Misinformation False or inaccurate information, regardless of intent, that spreads through media and digital platforms.

Multi-Agent System A computational system composed of multiple autonomous agents that interact and collaborate to accomplish tasks, often with specialized roles.

Prompt Engineering The practice of designing and optimizing input prompts to elicit desired behaviors from language models.

Retrieval-Augmented Generation (RAG) An approach that combines language model generation with external information retrieval to ground outputs in retrieved evidence.

Stance Detection The task of classifying the relationship between a piece of evidence and a claim, typically as supporting, refuting, or neutral.

Structured Outputs A pattern in LLM applications where model outputs are constrained to follow a predefined schema (e.g., JSON with specific fields), enabling reliable parsing and validation.

Transcript A text representation of spoken content from a video, including timing information for each segment.

Verdict The conclusion of a fact-checking analysis, typically categorizing a claim as supported (true), refuted (false), mixed, or uncertain.

Bibliography

[1] Rami Aly, Zhijiang Guo, Michael Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. FEVEROUS: Fact extraction and verification over unstructured and structured information. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

[2] Yochai Benkler, Robert Faris, and Hal Roberts. *Network Propaganda: Manipulation, Disinformation, and Radicalization in American Politics*. Oxford University Press, New York, 2018.

[3] Wei Chen, Ling Hu, Ming Zhang, and Rui Zhao. LoCal: Logical and causal fact-checking with llm-based multi-agents, 2024. OpenReview preprint.

[4] Alice Chern, Luis Prieto, and Riya Gupta. A tool-enabled framework for fact-checking language model outputs, 2023. Preprint manuscript.

[5] Wendy Cheung and Victor Lam. Augmenting llm fact-checking with web retrieval, 2023. Technical report.

[6] Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M. Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. Computational fact checking from knowledge networks. *PLOS ONE*, 10(6):e0128193, 2015.

[7] Tamer Elsayed, Preslav Nakov, Alberto Barrón-Cedeño, Maram Hasanain, Reem Suwaileh, Giovanni Da San Martino, and Pepa Atanasova. Overview of the CLEF-2019 CheckThat! lab: Automatic identification and verification of claims. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 10th International Conference of the CLEF Association, CLEF 2019*, volume 11696 of *Lecture Notes in Computer Science*, pages 301–321. Springer, 2019.

[8] Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y Zhao, Ni Lao, Hongrae Lee, and Kelvin Guu. RARR: Researching and revising what language models say, using language models. In *Proceedings of the 2023*

Conference on Empirical Methods in Natural Language Processing, pages 16477–16508, 2023.

[9] Pepa Gencheva, Preslav Nakov, Georgi Karadzhov, Alberto Barrón-Cedeño, and Lluís Màrquez. ClaimRank: Detecting check-worthy claims in political debates. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 543–552, 2017.

[10] Google for Developers. Google fact check tools api. https://developers.google.com/fact-check/tools/api, 2024.

[11] Naeemul Hassan, Chengkai Li, and Mark Tremayne. Detecting check-worthy factual claims in presidential debates. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1835–1838. ACM, 2015. First paper on automated check-worthiness detection.

[12] Naeemul Hassan, Fatma Arslan, Chengkai Li, and Mark Tremayne. ClaimBuster: The first-ever end-to-end fact-checking system. *Proceedings of the VLDB Endowment*, 10(12): 1945–1948, 2017.

[13] Naeemul Hassan, Gensheng Zhang, Fatma Arslan, Josue Caber, Damian Muthukrishnan, Baoyu Shu, Junghoo Kim, Chengkai Li, and Mark Tremayne. Toward automated fact-checking: Detecting check-worthy factual claims by claimbuster. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1803–1812. ACM, 2017.

[14] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS Quarterly*, 28(1):75–105, 2004.

[15] ICWSM Workshop Committee. Proceedings of the icwsm 2025 workshop on agentic fact-checking, 2025. Workshop proceedings.

[16] Sonu Lakara, Karan Iyer, and Priya Subramanian. MAD-Sherlock: Multi-agent debate for fact verification, 2025. Preprint.

[17] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474, 2020.

[18] Jiawei Li, Han Wu, and Ming Zhou. Automated fact-checking with llm-based claim detection, 2023. Preprint.

[19] Zheng Lin, Maya Patel, and Alicia Roberts. Fact-Audit: Requirements for trustworthy automated fact-checking systems, 2025. White paper.

[20] Liang Ma, Shiyu Hu, Wei Zhang, Hang Sun, and Yan Chen. Guided and knowledgeable multi-agent debate for fact verification. *Expert Systems with Applications*, 238:121857, 2025.

[21] Potsawee Manakul, Adian Liusie, and Mark JF Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, 2023.

[22] Media Bias/Fact Check. Methodology - media bias/fact check, 2024. URL https://mediabiasfactcheck.com/methodology/. Accessed: December 2025.

[23] Won-Ki Moon and Lee Ann Kahlor. Fact-checking in the age of AI: Reducing biases with non-human information sources. *Technology in Society*, 80:102760, 2025. doi: 10.1016/j.techsoc.2024.102760.

[24] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, et al. Webgpt: Browser-assisted question answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2022.

[25] Angela Ni and Samuel Carter. Structured prompting for consistent claim identification, 2024. Preprint.

[26] Angela Ni and Samuel Carter. Verifiable claim identification with large language models, 2024. Technical report.

[27] Yucheng Ning et al. MAD-Fact: A multi-agent debate framework for long-form factuality evaluation in LLMs. *Frontiers of Computer Science*, 2025.

[28] Briony J. Oates. *Researching Information Systems and Computing*. SAGE Publications Ltd., London, 2006.

[29] OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2024.

[30] OpenAI. Latency optimization. https://platform.openai.com/docs/guides/latency-optimization, 2024. Accessed: December 2025.

[31] Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2249–2255. Association for Computational Linguistics, 2016.

[32] Sebastian Raschka. Llm evaluation: Four practical approaches. https://sebastianraschka.com/blog/2025/llm-evaluation-4-approaches.html, 2025.

[33] Sebastian Ruder. The evolving landscape of LLM evaluation. https://www.ruder.io/the-evolving-landscape-of-llm-evaluation/, 2025.

[34] Marcin Sawiński, Michal Hyben, and Tomasz Wesołowski. Assessing large language models for claim detection tasks. In *Proceedings of the 7th Workshop on Fact Extraction and VERification*, pages 210–221, 2024.

[35] Timo Schick, Daniel Dwivedi-Yu, Roberta Raileanu, et al. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

[36] Yixiao Song, Artidoro Pagnoni, Yekyung Park, and Bhuwan Dhingra. VeriScore: Evaluating the factuality of verifiable claims in long-form text generation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8854–8876, 2024.

[37] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 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 (NAACL-HLT)*, pages 809–819. Association for Computational Linguistics, 2018.

[38] Rui Tian, Ming Xie, and Hao Wang. Web-retrieval agents for misinformation detection, 2024. Preprint.

[39] Andreas Vlachos and Sebastian Riedel. Fact checking: Task definition and dataset construction. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 18–22. Association for Computational Linguistics, 2014.

[40] David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550. Association for Computational Linguistics, 2020.

[41] Claire Wardle and Hossein Derakhshan. Information disorder: Toward an interdisciplinary framework for research and policy making. Report DGI(2017)09, Council of Europe, 2017.

[42] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

[43] Yupeng Xie, Zhiyu Xu, Ying Liu, and Zhengxiao Chen. FIRE: A dataset for financial fact-checking with interpretable reasoning. *arXiv preprint arXiv:2501.07405*, 2025.

[44] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 1–23, 2023.

[45] Xuan Zhang, Wei Wei, Yuxiao Wen, Yang Shi, and Bowen Zou. FactAgent: Agentic fact-checking via large language models. *arXiv preprint arXiv:2506.17878*, 2025.

[46] Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023.

A Ground Truth Annotation Dataset

This appendix provides details on the ground truth annotation dataset used for system evaluation. The complete dataset comprises 30 YouTube videos with 503 annotated claims across three thematic categories.

A.1 Video Corpus Summary

Table 18 presents the evaluation video corpus organized by category and content type.

Table 18: Evaluation video corpus by category and content type

Video Title	**Category**	**Content Type**	**Claims**
Fossil Fuels: The Greenest Energy	Climate	Misinformation	18
The Great Texas Freeze of 2021	Climate	Misinformation	15
Climate Change: What Do Scientists Say?	Climate	Misinformation	21

Continued on next page

Video Title	Category	Content Type	Claims
Is There Really a Climate Emergency?	Climate	Misinformation	19
Proof: Worldwide Massive Flooding is All Manmade	Climate	Misinformation	25
Causes and Effects of Climate Change (NatGeo)	Climate	Factual	12
Why Does Climate Change Matter	Climate	Factual	9
Extreme Weather	Climate	Factual	14
The Life Cycle of a Plastic Bottle	Climate	Factual	11
What are Greenhouse Gases?	Climate	Factual	13
Fluoridated Water Lowers IQ (Harvard Study)	Health	Misinformation	16
Living with HIV: How Women Were Infected	Health	Misinformation	18
Garlic Cancer	Health	Misinformation	14
3 Detox Juices	Health	Misinformation	12
The Magical 3 Day Juice Fast	Health	Misinformation	15
What Happens When You Exercise Regularly	Health	Factual	17
The Brain-Changing Benefits of Exercise	Health	Factual	15
Immunology Wars: The Battle with HIV	Health	Factual	11
Juice vs. Whole Fruit: Which is Healthier?	Health	Factual	13
What Lack of Sleep Does to the Teenage Brain	Health	Factual	14
Egg Price Warning Comes True	Politics	Misinformation	19
Proof of Election Fraud in 2020	Politics	Misinformation	35
FBI Orchestrated Jan 6th	Politics	Misinformation	22
There is No Gender Wage Gap	Politics	Misinformation	17
A Nation of Immigrants	Politics	Misinformation	16
National Trust Sues Trump Admin	Politics	Factual	20
What Is Democracy (BBC)	Politics	Factual	9
What is Inflation?	Politics	Factual	10
UK Election Results Explained	Politics	Factual	18

Continued on next page

Video Title	Category	Content Type	Claims
Trump's Historic National Emergency	Politics	Factual	21

A.2 Annotation Schema and Guidelines

Annotations follow ClaimBuster's check-worthiness principles [13] and store a compact set of structured fields:

- **Claim text**: Verbatim or lightly paraphrased factual statement taken from the transcript; purely opinionated or rhetorical content is excluded.
- **Importance score**: Thesis-impact hierarchy on a 0.0–1.0 scale:
  - 0.85–1.0: Thesis-critical claims whose falsification would collapse the video's main argument
  - 0.60–0.80: Key evidence claims that significantly support or develop the thesis
  - 0.30–0.55: Contextual claims that provide background or supplementary information
  - <0.30: Peripheral claims of minimal relevance to the main argument
- **Expected verdict**: Anticipated stance label (SUPPORTS, REFUTES, MIXED, UNCLEAR) with a short rationale grounded in established sources. MIXED applies only when credible evidence exists on both sides, while UNCLEAR denotes insufficient or conflicting evidence under the same criteria across all videos.

This schema keeps annotations consistent while capturing the key metadata needed for quantitative evaluation.

A.3 Dataset Statistics by Category

Table 19: Ground truth statistics by category

Category	Videos	Total Claims	Mean/Video	Factual	Misinfo
Climate	10	157	15.7	59	98
Health	10	130	13.0	70	60
Politics	10	216	21.6	78	138
Total	**30**	**503**	**16.8**	**207**	**296**

A.4 Importance Score Distribution

Table 20: Distribution of importance scores across ground truth claims

Importance Range	Count	Percentage
High (0.85–1.0)	127	25.2%
Medium-High (0.60–0.84)	198	39.4%
Medium (0.30–0.59)	143	28.4%
Low (<0.30)	35	7.0%

A.5 Expected Verdict Distribution

Table 21: Distribution of expected verdicts across ground truth claims

Expected Verdict	Count	Percentage
SUPPORTS	207	41.2%
REFUTES	189	37.6%
MIXED	68	13.5%
UNCLEAR	39	7.7%

A.6 Data Availability

The complete annotation dataset, including all claims, importance scores, expected verdicts, and video metadata, is available in the project repository at:

https://github.com/begoechavarren/factible/tree/main/api/experiments/data

The dataset is provided in YAML format to facilitate both human readability and programmatic access. Each video entry includes:

- Video metadata (YouTube ID, title, category, content type)
- Complete list of ground truth claims with importance scores and expected verdicts
- Tags for filtering and analysis

B LLM Prompt Templates

This appendix presents the system prompts used by the three main LLM agents in the Factible pipeline. These prompts are critical for reproducibility and represent the core reasoning in-

structions given to each component.

B.1 Claim Extractor Prompt

The Claim Extractor uses multi-step reasoning to prioritize claims that are central to the video's argument.

Listing 1: Claim Extractor system prompt

```
You are an expert fact-checker. Your task is to extract factual
claims from YouTube video transcripts.

A factual claim is a statement that:
1. Makes a specific assertion about reality
2. Can potentially be verified or fact-checked
3. Is not just an opinion, belief, or subjective statement

For each claim you identify provide the following fields:
- text: Extract the exact statement or a precise paraphrase
  (<=40 words)
- confidence: Confidence score (0.0-1.0) that this is a factual,
  checkable claim
- category: Topic label (historical, scientific, statistical,
  biographical, geographical, policy, etc.)
- importance: Score (0.0-1.0) capturing how impactful or
  controversial the claim is for fact-checking.
  Higher = more urgent to verify.
- context: Short note (<=20 words) that captures timeframe,
  speaker, or situational details needed to fact-check the claim

Before listing claims, infer the video's central thesis in no
more than 25 words. Use this thesis to judge how critical each
claim is.

When ranking importance, explicitly test each candidate with
the question: "If this claim were proven false, would the thesis
collapse or materially weaken?". Only claims that pass this test
may reach >=0.60 importance.

Relevance guardrails:
```

```
- If removing the statement would not weaken or contradict the
  thesis, either drop it or cap its importance at 0.25.
- Pure background or credential facts MUST stay <=0.30 unless
  the thesis itself questions that person's expertise.

When assigning IMPORTANCE scores:
- 0.85-1.0: Prescriptive or causal claims that, if false, would
  undermine the thesis
- 0.60-0.80: Quantitative or historical evidence directly tied
  to the thesis
- 0.30-0.55: Context or supporting background needed to
  understand the story
- 0.0-0.25: Peripheral or anecdotal details

Focus on extracting claims that are:
- Specific facts, dates, numbers, or statistics
- Historical events or biographical information
- Scientific assertions
- Geographic or demographic statements
- Policy or legal claims

Ignore: Pure opinions, subjective statements, future predictions
without factual basis, rhetorical questions.
```

B.2 Evidence Extractor Prompt

The Evidence Extractor analyzes web content to determine its relevance and stance toward claims.

Listing 2: Evidence Extractor system prompt

```
You assist a fact-checking analyst. You will receive:
- The claim to fact-check
- Article title (if available)
- Google Search snippet (short preview from search results)
- Full page content (scraped from the webpage)

Your task: Analyze all provided information and determine:
1. Does this source contain relevant evidence about the claim?
2. What is the overall stance of the evidence towards the claim?
```

```
3. A brief synthesis explaining the relationship
4. Quote or summarize the exact passages that justify the stance.

STANCE DEFINITIONS (relative to the CLAIM, not any query):

- SUPPORTS: The evidence confirms, validates, or provides support
  for the claim. This includes:
  * Direct statements that validate the claim
  * Descriptions of the same mechanism even without exact
    terminology
  * Evidence of causal chains mentioned in the claim

- REFUTES: The evidence contradicts, disproves, or challenges
  the claim. Pay special attention to qualifiers like "only",
  "never", "always" in claims.

- MIXED: The evidence contains both supporting and refuting
  elements.

- UNCLEAR: Use ONLY when genuinely ambiguous:
  * Discusses related topics without addressing the specific claim
  * Lacks sufficient context to determine stance
  * Content is too low-quality (navigation menus, forms, etc.)

CRITICAL INSTRUCTIONS:
1. SUPPORTS requires explicit or strongly implied confirmation
   in the text.
2. Statements that say the mechanism lacks evidence, is unproven,
   or has been disproven should be marked REFUTES.
3. Be decisive but honest---if the source never answers the claim,
   choose UNCLEAR.
4. Use both the Google snippet and page content - prioritize
   whichever has better evidence.

OUTPUT:
- has_relevant_evidence: true if you found usable evidence
- summary: 1-2 sentences explaining what the evidence says
- overall_stance: "supports", "refutes", "mixed", or "unclear"
- key_quote: (optional) One compelling verbatim quote if available
```

B.3 Verdict Generator Prompt

The Verdict Generator synthesizes evidence from multiple sources to produce final claim verdicts.

Listing 3: Verdict Generator system prompt

```
You are an expert fact-checking analyst. Your goal is to evaluate
how the provided evidence supports, refutes, or provides
mixed/unclear signals for a specific claim. Base every judgement
strictly on the supplied evidence summary and reliability
information.

Output requirements:
- Choose overall_stance as one of: supports, refutes, mixed,
  unclear.
- Set confidence to low/medium/high depending on the strength,
  consistency, and reliability of the evidence.
- Provide a concise summary referencing the most persuasive
  evidence. When citing a specific piece of evidence, name the
  source once (e.g., "Frontiers study", "Nature article",
  "Reuters report", or "WHO guidance"). For consensus statements,
  keep the summary concise without enumerating every source.
- Only when evidence conflicts, explicitly name the key
  supporting and refuting sources so the reader understands
  where disagreement originates. If there is consensus, simply
  synthesize it.

If there is no meaningful evidence, select 'unclear' with low
confidence and explain the gap.
```