POLITECNICO DI TORINO

Master's Degree in Computer Engineering
Artificial Intelligence and Data Analytics

Master's Degree Thesis

LLM-Based Fact-Checking with Retrieval and Agentic Workflows for Italian Fact-Checking Datasets

Supervisor

Prof. Andrea BOTTINO

Candidate

Jad KHARRAT

July 2026

Abstract

The rapid growth of online information and misinformation has increased the need for scalable fact-checking systems. Professional fact-checking remains essential, but it is difficult to scale to the volume and speed at which claims circulate online. This has motivated research on automated verification systems that can support the fact-checking process.

Fact-checking, however, is not a single uniform task. Claims about stable and widely documented facts can often be checked against consolidated knowledge, while political claims are more difficult. They are time-sensitive, context-dependent, and often require precise evidence about dates, institutions, actors, statistics, and public statements. For these claims, access to relevant and timely evidence is often necessary before a reliable verification decision can be made.

Retrieval-based fact-checking systems address this problem by searching for external evidence before making a prediction. Recent LLM-based approaches extend this idea by using language models not only as classifiers, but also as components that can interpret evidence, refine search queries, and support more structured verification workflows. At the same time, retrieval-based evaluation remains difficult: historical datasets can allow leakage through memorized model knowledge, original fact-checking articles, or sources published after the claim. Without realistic evaluation procedures, these effects can lead to overestimating system capabilities and misunderstanding what retrieval actually contributes.

This thesis investigates how web retrieval and agentic workflows can support LLM-based fact-checking under realistic evaluation constraints. It contributes a controlled comparison of multiple verification settings, a zero-shot error-repair strategy to isolate the contribution of retrieval, and a leakage-aware evaluation framework for realistic political fact-checking. The compared settings include claim-only verification, oracle evidence, plain web retrieval, heuristic web retrieval, and a controlled retrieval agent that refines the search process before verification.

Experiments are conducted on two Italian datasets with complementary roles. FEVER-IT is used as a controlled benchmark for studying the behavior of the retrieval pipeline, while CheckIT is used as a realistic political fact-checking setting with stricter evidence requirements and higher leakage risk. The results show that

evidence quality is the main driver of verification performance. Web retrieval repairs many claim-only errors, while the proposed retrieval agent provides a smaller but measurable improvement by refining the retrieval process. Additional model checks confirm that CheckIT remains difficult across different open models.

Overall, the thesis shows that trustworthy automated fact-checking depends not only on the reasoning ability of language models but also on evidence grounding, retrieval quality, and realistic evaluation procedures.

Table of Contents

Abstract I

1 Introduction 1
- 1.1 Motivation and Problem Definition . . . 1
- 1.2 Objectives of the Thesis . . . 2
- 1.3 Research Questions . . . 3
- 1.4 Contributions . . . 3
- 1.5 Thesis Structure . . . 4

2 Background and Related Work 5
- 2.1 Automated Fact-Checking . . . 5
- 2.2 Large Language Models . . . 6
- 2.3 Retrieval-Augmented Generation . . . 7
- 2.4 Controlled Retrieval Workflows . . . 7
- 2.5 Non-LLM Approaches . . . 8
- 2.6 Datasets . . . 9
  - 2.6.1 FEVER and FEVER-IT . . . 9
  - 2.6.2 CheckIT . . . 10
- 2.7 Chapter Summary . . . 10

3 Methods 11
- 3.1 Task Definition . . . 11
- 3.2 Baseline Systems . . . 13
  - 3.2.1 Zero-Shot Baseline . . . 13
  - 3.2.2 Oracle Baseline . . . 13
  - 3.2.3 Error Subset Generation . . . 14
  - 3.2.4 Plain Retrieval . . . 15
  - 3.2.5 Heuristic Retrieval . . . 16
- 3.3 Proposed System . . . 17
  - 3.3.1 Proposed Controlled Retrieval Workflow . . . 18
  - 3.3.2 Leakage-Safe Retrieval . . . 20

3.4 Experimental Configuration 21
3.4.1 Models and Inference Settings 21
3.4.2 Retrieval Providers 22
3.4.3 Retrieval and Proposed-System Parameters 22
3.4.4 Evaluation Metrics 23
3.4.5 Prompting Strategy 24
3.4.6 Implementation Details 25
3.5 Chapter Summary 25

4 Experimental Evaluation 26
4.1 FEVER-IT Evaluation 27
4.1.1 Zero-Shot Results 27
4.1.2 Oracle Results 28
4.1.3 Plain Retrieval Baselines 28
4.1.4 Heuristic Retrieval Results 29
4.1.5 Proposed System Results 30
4.2 Retrieval Provider Comparison 31
4.3 CheckIT Evaluation 33
4.3.1 Zero-Shot Results 34
4.3.2 Leakage-Safe Retrieval Results 34
4.3.3 Proposed System Results 36
4.4 Additional Model Checks 38
4.5 Error Analysis 39
4.6 Final Results Summary 40
4.7 Chapter Summary 41

5 Discussion and Conclusion 42
5.1 Discussion 42
5.1.1 Answer to RQ1 42
5.1.2 Answer to RQ2 43
5.1.3 Answer to RQ3 43
5.1.4 Broader Interpretation 44
5.2 Limitations 44
5.2.1 Threats to Validity 45
5.3 Future Work 45
5.4 Conclusion 46

A Additional Experimental Tables 48
A.1 FEVER-IT Main Results 48
A.2 Retrieval Provider Comparison 49
A.3 CheckIT Repair and Combined Results 49

A.4 Per-Label Breakdown for Main Runs 50
A.5 Additional Model Checks 51
A.6 Fixed Retrieval Parameters 51
A.7 Prompt Templates 52

Bibliography 53

Chapter 1

Introduction

1.1 Motivation and Problem Definition

The rapid spread of misinformation through online platforms and social media has increased the need for automated fact-checking systems that can support human verification workflows [1, 2]. Political misinformation is especially challenging because claims often involve evolving events, institutional context, numerical details, and implicit references that are difficult to verify automatically.

Recent advances in Large Language Models (LLMs) have improved the ability of language systems to perform reasoning and natural language understanding tasks [3, 4, 5]. However, relying only on the internal knowledge encoded in language models remains problematic for factual verification. LLMs may contain outdated information, hallucinate unsupported facts, fail on temporally sensitive claims, or produce overconfident predictions without reliable evidence.

For this reason, many retrieval-based fact-checking systems combine language models with external evidence retrieval pipelines. Instead of verifying claims only through parametric memory, these systems retrieve supporting or contradicting information from external sources and use that evidence during verification. In principle, this makes the verifier more grounded and allows the system to access information that may not be reliably stored in the model parameters.

At the same time, retrieval-based fact-checking introduces its own methodological problems. Retrieved evidence may be noisy, incomplete, temporally inconsistent, or only loosely related to the claim. In political fact-checking settings, another important issue is retrieval leakage: unrestricted web search may retrieve the original fact-checking article, a mirrored version of it, or a later page that already contains the answer. In that case, the system may appear to verify a claim when it is actually finding an answer page.

This thesis investigates retrieval-based approaches for automated fact-checking

on Italian-language benchmarks, with particular focus on FEVER-IT and CheckIT. FEVER-IT is used as a controlled development benchmark, while CheckIT is used as a more realistic political fact-checking setting with higher leakage risk. The work focuses on understanding how retrieval quality, evidence construction, leakage-aware evaluation, and controlled query refinement affect downstream verification performance.

The central motivation is not simply to build a more complex fact-checking agent. Instead, the thesis asks where performance improvements actually come from: the verifier's internal knowledge, the availability of external evidence, the quality of retrieved pages, heuristic evidence construction, leakage filtering, or iterative retrieval control.

1.2 Objectives of the Thesis

The main objective of this thesis is to design, evaluate, and analyze retrieval-based fact-checking pipelines for Italian-language settings, with a focus on controlled evaluation rather than on producing the most highly tuned system.

More specifically, the work aims to:

- evaluate how much external retrieval improves over claim-only verification,
- compare different retrieval providers and retrieval strategies,
- analyze the contribution of heuristic retrieval pipelines and the proposed controlled retrieval workflow,
- study the impact of leakage-aware retrieval constraints in political fact-checking evaluation,
- and identify the main failure modes that remain when LLM-based verifiers are combined with web evidence.

The thesis focuses particularly on the relationship between retrieval quality and downstream verification performance. Rather than treating retrieval as a separate information-retrieval task, the experiments evaluate retrieval as part of an end-to-end verification pipeline. This is important because retrieved evidence is only useful if it helps the verifier make a better label decision.

Another objective is to analyze whether iterative retrieval control provides meaningful improvements once strong retrieval and evidence-construction baselines are already in place. This comparison is deliberately conservative: the proposed system is evaluated against a strengthened heuristic retrieval baseline rather than only against a simple direct-search baseline.

The experimental evaluation is conducted on FEVER-IT and CheckIT, which together provide complementary Italian fact-checking settings. FEVER-IT supports controlled development and comparison, while CheckIT tests the same ideas in a more realistic political fact-checking setting.

1.3 Research Questions

The thesis is guided by the following research questions:

1. How well can an LLM verify claims without evidence?
2. Can web retrieval repair zero-shot errors?
3. Does the protocol work on realistic political claims?

These questions are connected by a common theme: the thesis evaluates fact-checking systems not only by final accuracy, but also by the source of the improvement. In particular, the work distinguishes between improvements caused by retrieval itself, improvements caused by better evidence construction, and improvements caused by controlled query refinement. Provider choice, leakage control, and error analysis are therefore treated as supporting methodological analyses rather than as separate top-level research questions.

1.4 Contributions

The main contributions of this thesis can be summarized as follows:

- a controlled comparison between claim-only verification, oracle evidence verification, plain web retrieval, heuristic web retrieval, and the proposed system under a shared label space;
- a zero-shot error-repair protocol for measuring whether retrieval can correct claims that the model fails to verify without external evidence;
- a leakage-aware retrieval protocol for political fact-checking, based on domain blocking, date filtering, and rejection logs;
- and an empirical analysis of retrieval quality, provider choice, controlled query refinement, and remaining error types on FEVER-IT and CheckIT.

The thesis additionally investigates the practical limitations of retrieval-based fact-checking systems in political verification scenarios, particularly regarding retrieval leakage, contextual ambiguity, weak metadata, numerical reasoning, temporal consistency, and evidence reliability.

The contribution is therefore mainly empirical and methodological. The thesis does not claim to produce a deployable fact-checking system or a state-of-the-art result. Instead, it provides a controlled analysis of what retrieval and the proposed workflow add to LLM-based verification, and where these systems still fail under more realistic evaluation constraints.

1.5 Thesis Structure

The remainder of this thesis is organized as follows.

Chapter 2 presents the background and related work relevant to automated fact-checking, including large language models, retrieval-augmented generation, agentic systems, non-LLM approaches, and Italian fact-checking datasets.

Chapter 3 describes the methodological framework adopted throughout the thesis. This includes the task definition, baseline systems, retrieval pipelines, heuristic retrieval strategies, the proposed controlled retrieval workflow, the leakage-aware retrieval protocol, and the experimental configuration.

Chapter 4 presents the experimental evaluation conducted on FEVER-IT and CheckIT. It reports the zero-shot and oracle baselines, retrieval-provider comparisons, heuristic and proposed-system results, leakage-aware CheckIT evaluation, additional model checks, and error analysis.

Finally, Chapter 5 discusses the main findings of the thesis, its limitations, and possible future research directions.

Chapter 2

Background and Related Work

2.1 Automated Fact-Checking

Automated fact-checking aims to assess the veracity of factual claims using computational methods and external evidence sources [1]. The task has received increasing attention in recent years because of the rapid spread of misinformation through online platforms, social media, and large-scale content generation systems [2].

Although fact-checking pipelines vary across datasets and applications, most automated fact-checking systems follow a common structure consisting of:

- claim identification,
- evidence retrieval,
- and claim verification [1].

The first stage identifies claims that are potentially verifiable or check-worthy. The second stage retrieves supporting or contradicting evidence from external sources such as web pages, databases, or knowledge graphs. The final stage evaluates whether the retrieved evidence supports, refutes, or fails to verify the original claim.

One of the most influential benchmarks for automated fact-checking is FEVER (Fact Extraction and VERification) [6], which formalized claim verification as a supervised classification task using evidence retrieved from Wikipedia. The FEVER framework strongly influenced later fact-checking datasets and retrieval-based verification pipelines, including the Italian-language benchmarks explored in this thesis.

Early automated fact-checking approaches relied mainly on traditional machine learning pipelines, stance detection, and natural language inference methods [1, 6].

More recent systems increasingly combine large language models with retrieval-augmented generation and tool-based workflows in order to improve reasoning capabilities and evidence utilization.

At the same time, realistic fact-checking remains challenging because verification performance depends not only on reasoning quality, but also on retrieval quality, temporal consistency, contextual interpretation, and evidence reliability. These challenges become visible in political fact-checking settings, where claims often depend on implicit context or evolving real-world information.

2.2 Large Language Models

Large Language Models (LLMs) are neural language models trained on large-scale text corpora using transformer-based architectures [3, 4]. Recent LLMs have demonstrated strong performance across a wide range of natural language processing tasks, including question answering, summarization, reasoning, and information extraction.

Modern instruction-tuned LLMs are relevant for fact-checking because they can perform claim verification in zero-shot or few-shot settings without requiring task-specific supervised training [4, 5]. Depending on the evaluation setting, the model may receive only the claim or the claim together with external evidence, then predict a verification label through generative reasoning.

However, relying only on the internal knowledge encoded in LLM parameters presents important limitations for factual verification. The model may contain outdated information, hallucinate unsupported facts, or fail when claims require precise numerical, temporal, or contextual reasoning [1, 7].

These limitations become problematic in political fact-checking scenarios, where claims often depend on recent events, institutional context, or evolving real-world information not reliably captured during model pretraining.

For this reason, recent research increasingly combines LLMs with external retrieval systems and retrieval-augmented generation pipelines [8, 7]. Instead of relying only on parametric memory, retrieval-based systems dynamically gather evidence from external sources and use that evidence during verification.

In this thesis, LLMs are used primarily as verifiers operating on either:

- claim-only input,
- gold evidence,
- or externally retrieved evidence.

The experiments focus mainly on understanding how retrieval quality and retrieval orchestration affect verification performance once an LLM verifier is already available.

2.3 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) combines language models with external retrieval systems in order to improve factual grounding and access to information beyond the model's internal parameters [8]. Instead of relying only on parametric memory, a RAG system dynamically retrieves relevant documents or evidence and conditions the model output on that retrieved information.

RAG-based approaches are relevant for automated fact-checking because many verification tasks depend on recent, domain-specific, or highly contextual information that may not be reliably encoded within a pretrained language model [8, 1].

A typical retrieval-augmented pipeline consists of:

- query generation,
- document retrieval,
- evidence selection,
- and evidence-conditioned generation or classification [8].

The retrieval stage is therefore a critical component of the overall system. Weak retrieval quality can expose the verifier to irrelevant, incomplete, or misleading evidence, even when the downstream language model itself is capable of correct reasoning.

Recent fact-checking systems increasingly integrate web retrieval into verification workflows in order to improve evidence coverage and factual grounding [8, 7]. However, open-web retrieval also introduces new challenges, including noisy evidence, metadata inconsistencies, retrieval leakage, and temporal unreliability. These issues are central to this thesis because the proposed system retrieves evidence from the open web rather than from a closed evidence collection.

In this thesis, the task is not open-ended generation but retrieval-augmented verification: retrieved evidence is used to support a label decision. The implementation combines web-search providers, evidence-construction pipelines, and leakage-safe filtering mechanisms. The experiments focus on understanding how retrieval quality, evidence construction, and controlled retrieval behavior affect downstream verification performance.

2.4 Controlled Retrieval Workflows

Tool-using and agentic LLM systems extend standard retrieval-augmented pipelines by introducing explicit workflow control, intermediate reasoning stages, and dynamic interaction between language models and external tools [7, 9, 10]. In the context of this thesis, the relevant form of agentic behavior is narrow and controlled: instead

of performing a single retrieval-and-verification step, the proposed workflow can iteratively refine queries, inspect intermediate evidence, and decide whether the evidence is sufficient for verification.

Recent agentic frameworks frequently combine language models with planners, tool-use modules, and retrieval systems in order to support more complex reasoning workflows [7, 9, 10]. These systems are relevant for fact-checking because verification often requires multiple retrieval attempts, evidence comparison, or contextual interpretation rather than a single search step.

Some agentic architectures also include memory mechanisms. In this thesis, however, persistent long-term memory is not implemented as an experimental component. The proposed workflow instead uses lightweight short-term state management through LangGraph[1], treated here as a software framework for implementing explicit state transitions. During a single verification run, the system stores:

- retrieved pages,
- URLs,
- refinement decisions,
- and evidence gathered across iterations.

This allows the workflow to refine weak retrieval attempts without discarding previously collected information.

The experiments therefore evaluate a constrained form of agentic behavior focused on retrieval rather than on open-ended autonomy. The main functions are:

- query planning,
- evidence sufficiency evaluation,
- iterative retrieval,
- and retrieval control under leakage-safe constraints.

2.5 Non-LLM Approaches

Before the widespread adoption of large language models, automated fact-checking systems relied mainly on traditional machine learning pipelines, stance detection methods, and natural language inference architectures [1, 6].

Many earlier approaches treated fact-checking as a supervised classification task using manually engineered features, lexical overlap measures, TF-IDF representations, or sentence-level similarity metrics [1]. These systems often depended heavily on structured datasets and fixed retrieval pipelines.

[1]LangGraph is an open-source Python library for building stateful multi-step LLM workflows: `https://langchain-ai.github.io/langgraph/`.

Stance detection and natural language inference (NLI) became particularly influential in retrieval-based verification settings [1, 6]. In these approaches, the system first retrieves candidate evidence and then predicts whether the evidence supports, contradicts, or is unrelated to the claim.

Traditional machine learning models were also explored in Italian fact-checking research. For example, the CheckIT work evaluated Support Vector Machine (SVM) baselines using TF-IDF-based representations in order to estimate whether claim veracity could be predicted directly from claim text alone [2]. These claim-only baselines are useful in this thesis because they represent the opposite end of the evaluation spectrum from retrieval-augmented verification: they test whether surface lexical information is enough before external evidence is introduced. The limited performance reported for these baselines reinforces the difficulty of factual verification without reliable evidence, especially when claims are political, contextual, and time-sensitive.

Although recent research increasingly relies on LLM-based systems, many of the underlying challenges remain similar. Retrieval quality, evidence relevance, contradiction detection, and contextual interpretation continue to play a central role even when large language models are used as the final verifier.

2.6 Datasets

This thesis focuses on two Italian fact-checking datasets: FEVER-IT and CheckIT. They are used as complementary evaluation settings. FEVER-IT provides a more controlled benchmark for developing and comparing retrieval variants, while CheckIT provides a more realistic political fact-checking setting where leakage-aware retrieval becomes central.

2.6.1 FEVER and FEVER-IT

FEVER (Fact Extraction and VERification) is one of the most influential benchmarks for automated fact-checking [6]. The original dataset frames fact verification as a supervised classification task using claims paired with supporting or refuting evidence retrieved from Wikipedia.

FEVER-IT extends the FEVER framework to the Italian language [11]. In this thesis, FEVER-IT is used as the primary controlled development benchmark because it provides:

- cleaner labels,
- lower retrieval leakage risk,
- and a more stable environment for retrieval experimentation.

Most retrieval and proposed-system experiments in this thesis were first developed and evaluated on FEVER-IT before transfer to the more realistic political fact-checking setting.

2.6.2 CheckIT

CheckIT is an Italian political fact-checking dataset built from professionally verified claims collected from Pagella Politica [2]. Compared to FEVER-IT, CheckIT contains more context-dependent and politically grounded claims, making verification substantially more difficult.

The dataset is particularly relevant for evaluating realistic retrieval behavior because unrestricted web search can easily retrieve the original fact-checking articles or pages that already contain the answer. For this reason, CheckIT became the primary benchmark for leakage-safe retrieval evaluation in this thesis.

CheckIT also motivated several of the retrieval constraints and filtering strategies introduced in the proposed pipeline, including:

- domain filtering,
- temporal filtering,
- and contradiction-aware evidence handling.

2.7 Chapter Summary

This chapter introduced the concepts needed for the experimental design: automated fact-checking, LLM-based verification, retrieval-augmented verification, controlled query refinement, non-LLM baselines, and the two datasets used in the evaluation. The next chapter turns these concepts into the concrete baseline systems, proposed workflow, leakage controls, and fixed experimental configuration.

Chapter 3

Methods

3.1 Task Definition

This thesis studies automated fact-checking as a claim verification problem [6, 1]. Given a textual claim, the system predicts whether the claim is supported, refuted, or not verifiable from the information available to the model. The experiments compare two broad sources of information: the parametric knowledge already contained in a Large Language Model (LLM), and external evidence retrieved from the web [8, 7].

The main experimental benchmark is FEVER-IT, which is used as the controlled development setting for retrieval and proposed-system comparisons. CheckIT is used as a more realistic transfer setting because its political claims are closer to real fact-checking workflows and carry a higher risk of retrieval leakage.

The experiments use a unified three-way label space:

- SUPPORTS: the evidence supports the claim.
- REFUTES: the evidence contradicts the claim.
- NOTENOUGHINFO: the evidence is insufficient to decide whether the claim is true or false.

The same label space is used for FEVER-IT and CheckIT by mapping the original dataset labels into the three verification labels above. This makes it possible to compare claim-only, retrieval-based, and proposed-system runs under a common evaluation format.

For CheckIT, the original verdict labels were mapped into this common label space as shown in Table 3.1. Labels outside the two clear truth-value categories were treated as insufficiently determinate for the three-way verification task. Ambiguous, empty, or undefined CheckIT verdicts were therefore collapsed into NOTENOUGHINFO rather than assigned to SUPPORTS or REFUTES, because they do not provide a reliable positive or negative truth-value target.

Table 3.1: CheckIT label mapping used in the experiments.

Original CheckIT verdict	Internal verification label
`Vero`	SUPPORTS
`Falso`	REFUTES
`Impreciso`	NOTENOUGHINFO
`not defined`, `nd`, empty, or unknown	NOTENOUGHINFO

Across the evaluated systems, the workflow can be summarized as:

```
claim
-> retrieval step, when used
-> evidence construction
-> verifier model
-> label
```

The main system variants are ordered from claim-only verification to increasingly controlled retrieval settings.

System	**Pipeline**	**What it measures**
Zero-shot	claim → LLM verifier → label	Model behavior without external evidence
Oracle evidence	claim + gold dataset evidence → LLM verifier → label	Upper bound when relevant evidence is already available
Plain web	claim → web search → evidence → LLM verifier → label	External evidence without stronger filtering and ranking controls
Heuristic web	claim → search → filter/rank → evidence block → LLM verifier	Strong non-agentic web-retrieval baseline
Proposed system	plan query → search/read → judge evidence → refine if needed → verify	Controlled retrieval workflow with query refinement

Figure 3.1: System variants compared in the thesis. The variants progress from claim-only verification to web retrieval, heuristic retrieval, and the proposed controlled retrieval workflow.

In claim-only settings, the retrieval and evidence-construction steps are omitted. In retrieval-based settings, the claim is used to search the web, candidate pages

are fetched and filtered, and the resulting evidence text is passed to the verifier together with the original claim. In the proposed-system setting, the workflow can additionally judge whether the first retrieval attempt is sufficient and refine the query before producing a final prediction.

A central methodological concern is retrieval leakage. For political fact-checking datasets, especially CheckIT, a web search can retrieve the original fact-checking article or a later article that already contains the answer. This is related to the broader problem of evaluation contamination in language-model benchmarks [2, 12, 13]. This would measure the ability to find an answer page rather than the ability to verify the claim from independent evidence. For this reason, the proposed pipeline includes domain filtering, date filtering, and rejection logging, described later in this chapter.

The rest of the chapter is organized around the distinction between baseline systems and the proposed system. The goal is not only to maximize accuracy, but to understand where performance comes from: model knowledge, retrieval quality, evidence construction, leakage control, or controlled query refinement.

3.2 Baseline Systems

The baseline systems establish the reference points used to interpret the proposed system. They include claim-only verification, oracle evidence verification, plain web retrieval, and heuristic web retrieval.

3.2.1 Zero-Shot Baseline

The zero-shot baseline receives only the claim text. No retrieved web pages, dataset evidence, or external knowledge source is provided.

```
claim
-> LLM verifier
-> label
```

Given an input claim, the verifier predicts one of the three target labels. During development, the main model was `gemma3:latest`, executed locally through Ollama with deterministic inference settings. This setting measures how far the model can go without retrieval and identifies the examples for which external evidence is needed.

3.2.2 Oracle Baseline

The oracle baseline removes the retrieval problem by giving the verifier the gold evidence associated with the claim in the dataset annotations.

```
claim + gold evidence
-> LLM verifier
-> label
```

This setting is not intended as a deployable system. Instead, it estimates how well the verifier can classify claims when relevant evidence is already available. The gap between zero-shot and oracle performance indicates how much improvement may be possible through better retrieval and evidence construction.

3.2.3 Error Subset Generation

Retrieval experiments are evaluated primarily on claims that the zero-shot baseline gets wrong. After running the zero-shot baseline, each example is assigned to one of two groups:

```
evaluation subset
-> zero-shot prediction
-> split into:
    - correct predictions
    - incorrect predictions
```

The incorrect predictions form an error subset. This makes retrieval evaluation more focused: rather than measuring performance on examples that the model already solved without evidence, it measures whether retrieval can repair previous mistakes.

In the FEVER-IT development experiments, the first 200 examples of the development split produced:

- 93 correct predictions
- 107 incorrect predictions

The resulting 107-example error subset was used as the controlled benchmark for plain retrieval, heuristic retrieval, and the proposed system. For combined scores, the original zero-shot correct predictions are kept and the retrieval system is applied only to the zero-shot error subset.

The first 200 examples were used as a controlled experimental subset rather than as a claim of full-dataset coverage. This choice made it possible to run repeated web-retrieval experiments under practical runtime and API constraints, inspect individual retrieval failures manually, and compare providers and system variants on the same fixed examples. The subset size is therefore a methodological compromise: large enough to reveal recurring failure modes, but small enough to keep retrieval and proposed-system experiments reproducible during development.

The same error-repair structure is later used for the CheckIT transfer evaluation, where the zero-shot stage produced 67 correct predictions and 133 errors.

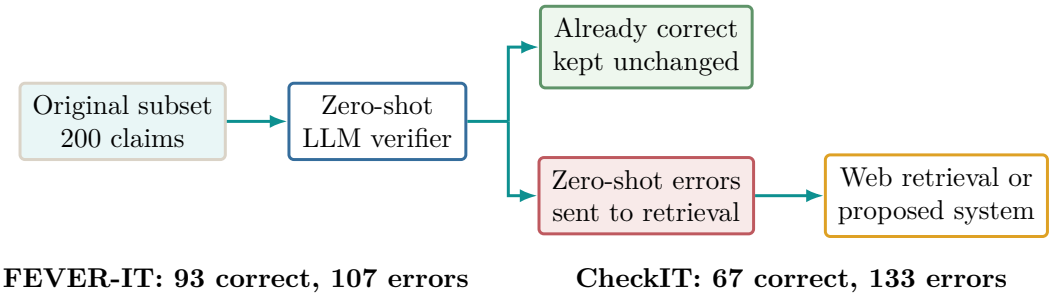

Figure 3.2: Zero-shot error-repair evaluation protocol. Retrieval is evaluated on examples that the claim-only verifier initially classified incorrectly, while already-correct zero-shot predictions are kept unchanged for combined scores.

3.2.4 Plain Retrieval

Retrieval-based fact-checking introduces externally retrieved web evidence into the claim verification pipeline. In this thesis, retrieval is evaluated as a repair mechanism for claims that the zero-shot baseline classified incorrectly. The retrieval component is therefore not treated as a separate information-retrieval task, but as part of an end-to-end verification system.

```
claim
-> web retrieval
-> page fetching and filtering
-> evidence construction
-> LLM verifier
-> label
```

For each claim, the system queries a web-search provider, fetches a limited number of candidate pages, converts the fetched text into an evidence block, and asks the verifier to classify the original claim using only that evidence. The default retrieval configuration requests up to eight search results and fetches up to four pages. Retrieved page text is truncated to approximately 12,000 characters per page, and the final evidence block is limited to approximately 5,000 characters.

Three search providers were used during development: Serper, Tavily, and Exa. Serper was used in early experiments, while Tavily and Exa became the main controlled comparison providers. Exa was selected as the final provider for the strongest heuristic baseline and the proposed system because it produced the

strongest heuristic web result and better date metadata coverage in the controlled FEVER-IT experiments.

The plain web baseline uses a direct retrieval-and-verification pipeline. The claim text is used as the search query, the provider returns ranked candidate pages, and the system fetches text from the first usable results.

```
claim
-> search provider
-> candidate pages
-> fetched text
-> evidence block
-> verifier
-> label
```

In the clean version of this baseline, evidence construction is intentionally simple: the system uses the beginning of the fetched page text, with minimal claim-specific sentence selection. This provides a useful control condition because it measures the effect of external retrieval without adding many task-specific heuristics.

The plain retrieval baseline is important for provider comparison. By keeping the downstream verifier and evidence format fixed, differences between Tavily and Exa can be interpreted primarily as differences in retrieval quality, metadata availability, and the relevance of returned snippets and pages.

3.2.5 Heuristic Retrieval

The heuristic retrieval mode was introduced after early runs showed a recurring failure mode: retrieved pages were often topically related to the claim, but did not contain the decisive factual detail needed for verification. This frequently caused the verifier to over-predict SUPPORTS.

```
claim
-> web retrieval
-> domain and quality ranking
-> snippet selection
-> evidence construction
-> guarded verification
-> label
```

The heuristic mode modifies both retrieval ordering and evidence construction. Search results from higher-quality domains are prioritized, while low-value or hard-blocked domains are avoided. The fetched text is then scored at sentence level using claim-term overlap, named-entity overlap, numeric and date overlap, and

contradiction cues. The evidence block is built from the most relevant snippets rather than simply from the beginning of each page.

The verifier is also protected by lightweight post-processing guards. These guards check whether a SUPPORTS prediction is missing important claim numbers, named entities, or relation-specific anchors such as family relations, protagonists, capitals, or exclusivity terms. Additional checks handle death/life-status claims, alternative numeric details, and contradiction cues. When decisive details are missing, the system can downgrade an overconfident SUPPORTS or REFUTES prediction to NOTENOUGHINFO.

Table 3.2 summarizes the main heuristic components. These heuristics are fixed rules used to build a stronger non-agentic retrieval baseline; they are not learned parameters and were not tuned through a full hyperparameter search.

Table 3.2: Summary of heuristic retrieval components.

Heuristic	Purpose
Domain/source ranking	Prioritize more useful or reliable sources before page fetching
Claim-term overlap	Improve topical relevance between the claim and retrieved snippets
Entity overlap	Preserve factual anchors such as people, places, organizations, and titles
Number/date overlap	Improve precision for claims involving quantities, years, dates, and comparisons
Contradiction cues	Help detect refutations when evidence contains explicit incompatibility signals
Weak-evidence fallback	Return NOTENOUGHINFO when retrieved evidence is too generic or incomplete
Numeric/date guard	Reduce false SUPPORTS predictions caused by partial matches or missing decisive numbers

This heuristic pipeline is the strongest non-agentic baseline in the thesis. It is also the baseline that the proposed system must improve over, which makes the comparison stricter than comparing the proposed system only against a simple plain retrieval setup.

3.3 Proposed System

The proposed system builds on the strongest heuristic retrieval baseline by adding controlled query refinement and leakage-safe filtering. The additional control is intended to improve retrieval, while the leakage-safe protocol makes evaluation

more realistic for political fact-checking datasets.

3.3.1 Proposed Controlled Retrieval Workflow

The proposed system tests whether iterative retrieval control improves over the strongest non-agentic heuristic baseline. It uses the same verifier and retrieval components as the heuristic pipeline, but adds a LangGraph workflow that can plan a query, inspect the retrieved evidence, decide whether the evidence is sufficient, and refine the query when needed. This design is motivated by the broader tool-use and reasoning-action literature, especially ReAct-style reasoning with external actions and Toolformer-style tool use [9, 10].

```
claim
-> planner
-> search_read
-> judge
-> refine_query, if evidence is insufficient
-> verify or give_up
```

The important difference from heuristic retrieval is not a different verifier, but the ability to judge evidence sufficiency and refine the search query before verification.

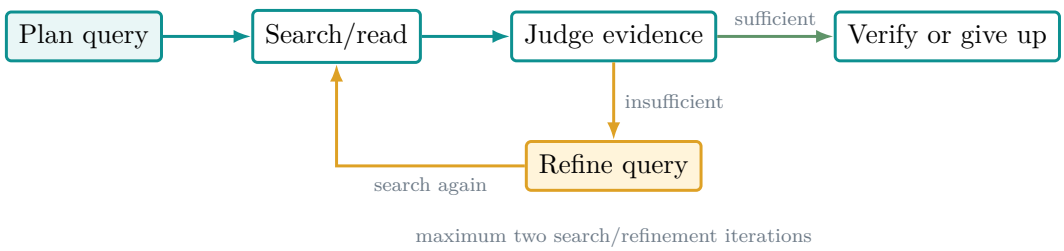

Figure 3.3: Proposed controlled retrieval workflow. The workflow refines the query only when the current evidence is judged insufficient, while keeping the verifier and retrieval provider fixed.

Table 3.3: Nodes in the proposed controlled retrieval workflow.

Node	Role
Planner	Generate the initial factual web-search query from the claim
Search/Read	Retrieve candidate pages, apply filtering, fetch page text, and construct evidence
Judge	Decide whether the current evidence directly addresses the decisive factual relation
Refine Query	Rewrite the query when evidence is generic, weak, or missing factual anchors
Verify	Produce the final SUPPORTS, REFUTES, or NOTENOUGHINFO label from accumulated evidence
Give Up	Return NOTENOUGHINFO when useful evidence is still not found after the iteration budget

The planner node generates the initial search query from the claim. The search_read node executes retrieval, applies the same filtering and evidence-construction logic used by the heuristic baseline, and accumulates pages and URLs across iterations. The judge node then evaluates whether the evidence directly addresses the decisive factual relation in the claim.

The judge is instructed to mark evidence as sufficient only when it directly supports or directly contradicts the decisive part of the claim. Evidence is treated as insufficient when it only discusses the general topic or misses crucial names, dates, numbers, roles, relations, or other claim-specific anchors. A direct contradiction is therefore considered sufficient evidence for verification, while weak associations and partial topical overlap trigger query refinement.

If the evidence is weak, generic, or missing important factual anchors, the workflow enters the refine_query node. This node generates a more specific query using both the original claim and the judge's reason for insufficiency. The next search_read step appends newly retrieved evidence rather than replacing all previous evidence.

The final verifier predicts SUPPORTS, REFUTES, or NOTENOUGHINFO using the accumulated evidence. A give_up fallback returns NOTENOUGHINFO when the maximum iteration budget is exhausted and the evidence remains too weak. In the final configuration used for the controlled comparison, the proposed system uses Exa as the provider and allows at most two retrieval iterations.

This design is deliberately conservative. The agent is not allowed to browse indefinitely or rely on unconstrained reasoning outside the evidence block. Its contribution is limited to query planning, sufficiency judging, evidence accumulation,

and deciding when to stop. This keeps the comparison with the heuristic baseline focused on retrieval control rather than on changes to the verifier or label space.

3.3.2 Leakage-Safe Retrieval

One of the main concerns in retrieval-based fact-checking evaluation is retrieval leakage. In political fact-checking datasets such as CheckIT, a web search may retrieve the original fact-checking article, a mirrored article, or a later page that already contains the correct answer. If such pages are used as evidence, the evaluation no longer measures claim verification under realistic conditions.

To reduce this issue, the retrieval pipeline applies leakage-safe filtering before evidence construction. The protocol combines domain filtering, temporal filtering, and rejection logging.

The filtering pipeline is:

```
claim
-> retrieval
-> domain filtering
-> date filtering
-> evidence construction
-> verifier
-> label
```

The leakage-safe flow blocks direct fact-checking sources, filters by publication date when possible, and records why candidates are rejected.

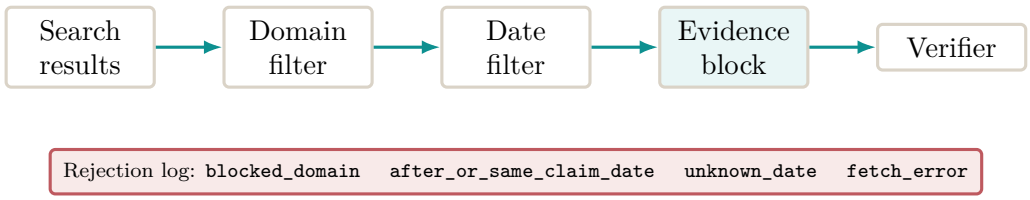

Figure 3.4: Leakage-safe retrieval protocol used for CheckIT. Domain filtering, date filtering, and rejection logs reduce direct answer-page lookup and make filtering behavior auditable.

The domain filter blocks known Italian fact-checking sources and related domains. In the current implementation, the global blocklist includes `pagellapolitica.it`, `facta.news`, `open.online`, and `bufale.net`, including their `www` subdomains. This prevents the verifier from directly reading professional fact-checking pages for the same or closely related claims.

The date filter uses the claim date when it is available. Retrieved pages with publication dates on or after the claim date are rejected, because they may contain information that was not available at the time the claim was made. If the search provider does not provide a publication date, the system attempts to infer one from the fetched page metadata. Unknown dates are allowed by default, but they are logged separately so that date coverage and filtering behavior can be analyzed.

Each rejection is recorded with a reason. The main logged reasons are:

- `blocked_domain`, for known fact-checking domains;
- `after_or_same_claim_date`, for pages published on or after the claim date;
- `unknown_date`, when the publication date cannot be determined;
- and `fetch_error`, when a candidate page cannot be opened or parsed.

These logs are included in the output JSONL files and are used to interpret retrieval failures, especially on CheckIT where leakage-safe filtering removes many high-ranking fact-checking pages.

This protocol cannot guarantee perfect leakage prevention, because open-web results may contain indirect summaries, syndicated material, or pages without reliable dates. However, it makes the evaluation stricter and more transparent than unrestricted web retrieval, and it avoids the most direct form of answer-page leakage.

3.4 Experimental Configuration

The experimental setup was designed to keep the verifier, retrieval parameters, and evaluation subsets as stable as possible while comparing different retrieval strategies. This was important because web retrieval is naturally variable: provider results can change over time, pages can fail to fetch, and generated queries can differ across proposed-system runs.

3.4.1 Models and Inference Settings

The primary language model was `gemma3:latest`, executed locally through Ollama. In the local Ollama installation used for the experiments, this tag corresponded to the `gemma3:4b` model, a 4-billion-parameter Gemma 3 variant with a 128K context window and text/image input support [14]. Only text input was used in the experiments. The main evaluation pipeline was standardized on this model so that differences between systems could be attributed mainly to retrieval and evidence construction rather than to model changes.

Verifier calls used deterministic-style settings, with temperature fixed to `0.0`. This does not make the whole system perfectly deterministic, because web search

and page retrieval remain external sources of variation, but it reduces unnecessary variability in the classification step.

Additional zero-shot and web-repair checks were later run with `llama3.1:8b`, `mistral:7b`, and `qwen2.5:7b`. These runs were used to test whether the CheckIT difficulty was specific to `gemma3:latest` or whether it also appeared across nearby open models. The use of locally hosted models avoided dependence on proprietary hosted LLM APIs for the verifier. External APIs were used only for web search providers.

3.4.2 Retrieval Providers

The retrieval systems used Serper, Tavily, and Exa at different stages of development. Serper was used as an early baseline, but later experiments focused on Tavily and Exa because they were more relevant for the controlled provider comparison.

Tavily was used extensively while refining the heuristic retrieval pipeline. It provided stable retrieval behavior, but date metadata coverage was limited in the recorded experiments. This made temporal filtering harder to apply consistently.

Exa was added later as a controlled provider comparison without changing the downstream evidence construction or verifier settings. It became the final provider for the strongest heuristic baseline and the proposed system because it produced the best FEVER-IT heuristic retrieval result and returned more publication-date metadata than Tavily.

This provider comparison is part of the method rather than only an implementation detail: retrieval quality strongly affects the evidence seen by the verifier, and therefore strongly affects the final label.

3.4.3 Retrieval and Proposed-System Parameters

The main retrieval and proposed-system parameters were fixed as practical defaults rather than optimized through a full hyperparameter search. This choice keeps the comparison focused on the evaluation protocol and on the difference between retrieval strategies, while avoiding excessive variation from repeated web-search calls.

The final configuration used the following settings:

Table 3.4: Fixed retrieval and proposed-system parameters.

Parameter	Value	Purpose
Search provider	Exa	Final provider after the FEVER-IT provider comparison; useful publication-date metadata
Main verifier	`gemma3:latest` / `gemma3:4b`	Local 4B verifier kept fixed for the main controlled comparison
Temperature	0.0	Reduces unnecessary variation in verifier predictions
`k_search`	8	Retrieves enough candidate results without making each query too expensive
`k_fetch`	4	Fetches a manageable number of pages for evidence construction
`page_max_chars`	12000	Limits each fetched page so long pages do not dominate the evidence block
`evidence_max_chars`	5000	Keeps the final evidence block large enough to be useful but small enough for the local verifier
`max_iterations`	2	Allows query refinement while limiting runtime, API calls, and instability

The value `max_iterations = 2` means that the workflow can perform an initial search and then refine the query when the judge finds the evidence insufficient. Most tuning during development focused on routing behavior, evidence sufficiency checks, contradiction handling, numerical and date guards, and provider choice rather than on a systematic sweep over iteration counts.

The character limits in Table 3.4 should be interpreted in the same way: they are fixed practical defaults selected to keep page text and evidence blocks manageable for local inference, not optimized limits discovered through exhaustive tuning.

3.4.4 Evaluation Metrics

The main evaluation metrics used throughout the experiments were Accuracy and Macro F1.

Accuracy measures the proportion of correctly classified claims over the total number of evaluated examples. Although accuracy provides an intuitive global measure of performance, it can be sensitive to label imbalance, particularly in datasets where certain labels occur more frequently than others.

For this reason, Macro F1 was additionally used as the primary balanced evaluation metric. Macro F1 computes the F1 score independently for each label and then averages the results across all labels, assigning equal importance to

SUPPORTS, REFUTES, and NOTENOUGHINFO.

In retrieval-based experiments, an additional repair accuracy metric is used. Repair accuracy measures the percentage of previously incorrect zero-shot predictions that become correct after retrieval and evidence augmentation. On the zero-shot error subset, repair accuracy is numerically equal to accuracy, but the name emphasizes the role of retrieval as an error-repair stage.

For combined evaluation settings, final scores were computed by combining:

- claims correctly classified during the initial zero-shot phase,
- and claims successfully corrected through retrieval-based verification.

This combined score is used for CheckIT to show the end-to-end effect of adding retrieval repair after the initial zero-shot prediction stage.

3.4.5 Prompting Strategy

The verification pipeline relies on task-specific prompts that constrain the output format and enforce evidence-grounded classification. The verifier is instructed to return valid JSON containing a label, reasoning, and sources.

For the verifier component, the model receives the input claim together with the retrieved evidence and is instructed to predict one of the target labels:

- SUPPORTS,
- REFUTES,
- NOTENOUGHINFO.

The prompts explicitly instruct the model to rely only on the provided evidence. This is especially important in retrieval-based settings: once evidence is provided, the goal is not to test the model's memory, but to test whether the retrieved evidence is sufficient for verification.

The proposed workflow further introduces specialized prompts for:

- query generation,
- evidence sufficiency evaluation,
- and query refinement.

The planner prompt generates concise factual search queries. The judge prompt evaluates whether the current evidence addresses the decisive factual detail of the claim. When evidence is insufficient, the refinement prompt generates a modified query using both the claim and the judge's reason for insufficiency.

Several heuristic constraints are also reflected in the verifier instructions, including contradiction-aware reasoning, numeric and temporal consistency, and fallback to NOTENOUGHINFO when evidence is weak or incomplete.

3.4.6 Implementation Details

The implementation is organized as a Python pipeline with separate modules for dataset loading, search providers, page fetching, retrieval filtering, evidence construction, verification, proposed-workflow execution, and evaluation.

Retrieval-related operations are handled by dedicated modules. The proposed workflow is implemented with LangGraph state transitions, while evaluation scripts write detailed per-example JSONL outputs. These outputs include the claim, gold label, predicted label, correctness, run configuration, retrieved URLs, evidence metadata, filtering counts, and rejection logs.

HTML content extraction uses BeautifulSoup-based preprocessing utilities. This removes many scripts, navigation fragments, and markup artifacts before evidence construction, but it does not guarantee clean article text. Boilerplate, cookie banners, table fragments, and encoding noise can still enter the evidence block, especially for pages with complex layouts or weak metadata. This extraction noise is treated as part of the practical retrieval setting and is considered again in the limitations. Additional custom components were implemented for:

- domain filtering,
- date filtering,
- evidence ranking,
- and retrieval logging.

This modular structure makes it possible to rerun the same evaluation subset with different providers or retrieval modes while keeping the verifier and output format fixed.

3.5 Chapter Summary

This chapter defined the methodological framework used in the thesis. It began with the baseline systems, including claim-only verification, oracle verification, plain retrieval, and heuristic retrieval. It then introduced the proposed system, which combines controlled query refinement with leakage-safe retrieval controls.

The key design choice is to evaluate retrieval as an error-repair mechanism: zero-shot failures form controlled subsets, and retrieval systems are compared by how many of those failures they correct. This makes the contribution of retrieval and controlled query refinement easier to analyze.

The next chapter reports the experimental results on FEVER-IT and CheckIT, including provider comparisons, heuristic-versus-proposed-system comparisons, leakage-safe transfer evaluation, additional model checks, and error analysis.

Chapter 4

Experimental Evaluation

This chapter presents the experimental evaluation of the retrieval-based systems and the proposed controlled retrieval workflow introduced in Chapter 3. The experiments are organized progressively, starting from controlled evaluation on FEVER-IT and later moving toward a more realistic political fact-checking setting using CheckIT.

The evaluation focuses on three main aspects:

- the effectiveness of retrieval augmentation compared to claim-only verification,
- the impact of retrieval quality and provider choice,
- and the contribution of the proposed workflow over strong heuristic retrieval baselines.

Particular attention is dedicated to leakage-safe retrieval evaluation, especially for political fact-checking datasets where unrestricted web retrieval can expose the system to already verified claims and answer pages.

The chapter first presents the controlled FEVER-IT experiments, followed by retrieval-provider comparisons, CheckIT transfer evaluation, additional model checks, and detailed error analysis.

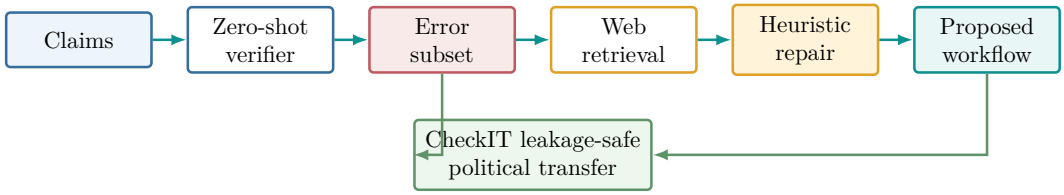

Figure 4.1: High-level experimental journey. The thesis first measures claim-only verification, then focuses retrieval experiments on zero-shot errors, compares heuristic and controlled retrieval repair, and transfers the protocol to leakage-safe political fact-checking on CheckIT.

4.1 FEVER-IT Evaluation

FEVER-IT was used as the primary controlled benchmark throughout system development. Compared to political fact-checking datasets such as CheckIT, FEVER-IT provides cleaner labels, lower immediate retrieval leakage risk, and a more stable environment for comparing retrieval strategies and the proposed workflow.

The experiments on FEVER-IT were designed to evaluate progressively more complex verification settings, starting from claim-only zero-shot verification and later introducing oracle evidence, retrieval augmentation, heuristic retrieval pipelines, and the proposed workflow.

Most controlled experiments were conducted on the first 200 examples of the FEVER-IT development split. This subset was selected to keep retrieval runtime and external search costs manageable, allow manual inspection of individual retrieval failures, and maintain a fixed comparison set across providers and system variants. The results should therefore be interpreted as controlled subset results rather than as full-dataset estimates.

The FEVER-IT experiments also served as the basis for generating the controlled zero-shot error subsets later used for retrieval-based evaluation. This made it possible to analyze retrieval quality more directly on claims that the verifier initially failed to classify correctly without external evidence.

4.1.1 Zero-Shot Results

The first experiment evaluates the claim-only zero-shot baseline on FEVER-IT without external retrieval or evidence augmentation.

The evaluation was conducted on the first 200 examples of the FEVER-IT development split using `gemma3:latest` as the verifier model.

The zero-shot baseline achieved:

- Accuracy: 0.465
- Macro F1: 0.462

These results indicate that the model already contains a moderate amount of factual knowledge relevant to the benchmark. However, the verifier still fails on a large portion of the claims, particularly when the claim depends on precise factual details, numerical information, or context not reliably encoded in the model parameters.

The zero-shot setting therefore establishes the initial claim-only reference point and serves as the starting point for all later retrieval-based experiments.

4.1.2 Oracle Results

The oracle experiment evaluates the verifier when gold evidence from the dataset annotations is directly provided together with the claim.

The evaluation was conducted on the same 200-example FEVER-IT subset used for the zero-shot baseline.

The oracle baseline achieved:

- Accuracy: 0.625
- Macro F1: 0.589

Compared to the zero-shot setting, the oracle configuration produces a substantial improvement in both Accuracy and Macro F1. This confirms that a large portion of the remaining verification errors are not caused only by weaknesses in the verifier itself, but also by the absence of reliable external evidence in the claim-only setting.

At the same time, oracle performance remains far from perfect even when correct evidence is available. Manual inspection of the outputs showed that some failures were related to numerical inconsistencies, implicit contradictions, or reasoning errors during evidence interpretation.

The gap between zero-shot and oracle performance therefore provides an approximate estimate of the potential benefit obtainable through successful retrieval and evidence construction.

4.1.3 Plain Retrieval Baselines

After establishing the zero-shot and oracle reference points, the next experiments evaluate retrieval-based verification using direct web retrieval without advanced heuristic filtering or iterative query refinement.

The retrieval experiments were conducted primarily on the 107-example FEVER-IT error subset generated from the initial zero-shot evaluation. This isolates claims that the verifier failed to classify correctly without external evidence.

Three retrieval providers were evaluated during development:

- Serper
- Tavily
- Exa

Early Serper experiments produced relatively weak retrieval quality and unstable metadata behavior. Later experiments focused mainly on Tavily and Exa, which provided more stable retrieval performance and better support for evidence filtering.

The strongest clean retrieval configuration was obtained using Tavily, which achieved:

- Accuracy: 0.411
- Macro F1: 0.407

Although retrieval improves several previously incorrect predictions, the plain retrieval baseline still suffers from a major limitation: retrieved evidence is often topically related to the claim without containing the decisive factual information required for reliable verification.

This issue was particularly visible for claims involving:

- numerical comparisons,
- temporal reasoning,
- and implicit factual contradictions.

These observations motivated the later introduction of heuristic evidence-ranking strategies and controlled query refinement.

4.1.4 Heuristic Retrieval Results

The heuristic retrieval pipeline was introduced to address the main weaknesses observed in the plain retrieval baseline, particularly cases where the retrieved evidence was relevant to the topic of the claim but insufficient for reliable verification.

Compared to the plain retrieval setting, the heuristic pipeline modifies both retrieval ordering and evidence construction. Retrieved snippets are ranked using claim overlap, named entities, numerical anchors, temporal information, and contradiction cues, while additional filtering guards attempt to reduce weak or misleading evidence.

The strongest heuristic retrieval configuration was obtained using Exa and achieved:

- Accuracy: 0.486
- Macro F1: 0.479

Since this evaluation was conducted on the 107-example FEVER-IT zero-shot error subset, this corresponds to repairing 52 previously incorrect predictions.

This result represents a clear improvement over the strongest plain retrieval baseline. The improvement was especially visible on claims where the decisive factual detail depended on:

- dates,
- quantities,
- named entities,
- or explicit contradiction signals.

However, several limitations remained even after heuristic filtering. In some cases, the retrieval stage still produced evidence that was strongly related to the claim topic without directly resolving the factual relation required for verification. The verifier also continued to over-predict SUPPORTS when evidence contained partial overlap with the claim but lacked decisive contradictory details.

These remaining weaknesses motivated the transition toward iterative retrieval control.

4.1.5 Proposed System Results

The final FEVER-IT experiments evaluate the proposed retrieval system implemented with LangGraph. The goal of the proposed system is not to replace the heuristic retrieval pipeline entirely, but to improve retrieval control through iterative query refinement and evidence sufficiency evaluation.

The proposed system uses the same verifier, retrieval provider, and evidence-construction logic as the heuristic baseline, while adding planner, judge, and query-refinement stages.

The strongest proposed-system configuration used Exa as the retrieval provider and achieved:

- Accuracy: 0.495
- Macro F1: 0.488

This corresponds to repairing 53 of the 107 FEVER-IT zero-shot errors.

Compared to the strongest heuristic retrieval baseline, the improvement is one repaired claim: 53 repaired cases instead of 52. This is a small descriptive difference on a 107-example subset, and it should not be interpreted as a statistically significant gain.

Manual inspection of the outputs suggests that the main benefit of the proposed system comes from improved retrieval behavior on difficult claims where:

- the first query retrieves only generic evidence,
- the decisive factual relation is implicit,
- or the initial retrieval attempt misses important named entities or numerical anchors.

At the same time, the experiments also showed that stronger retrieval orchestration alone is not sufficient to solve all verification failures. Some errors persisted even after multiple retrieval iterations because the retrieved evidence remained incomplete, ambiguous, or weakly connected to the factual relation expressed in the claim.

Taken together, the FEVER-IT experiments show that retrieval quality and evidence construction remain the dominant factors in verification performance, while the proposed system provides a small measured extension once a strong heuristic retrieval baseline is already in place. In the saved comparison against the Exa heuristic baseline, the proposed system fixed 8 baseline errors but also broke 7 cases that the baseline had classified correctly, reinforcing the need to interpret the FEVER-IT proposed-system result cautiously.

The FEVER-IT accuracy summary includes both full-subset scores and error-subset repair scores. Zero-shot and oracle are measured on the full 200-example subset, while heuristic retrieval and the proposed system are measured on the 107 zero-shot errors.

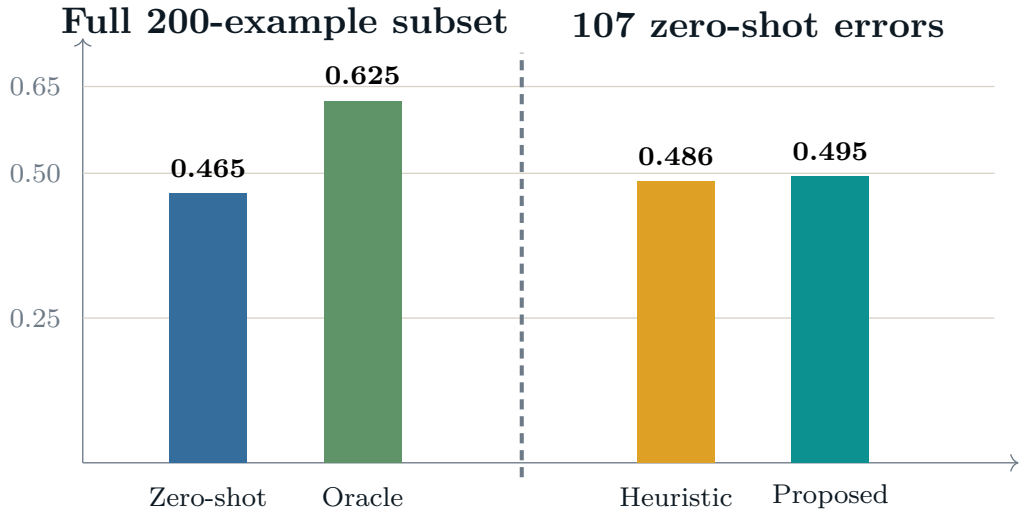

Figure 4.2: FEVER-IT accuracy summary. The left group reports full-subset scores on 200 examples; the right group reports repair accuracy on the 107 zero-shot errors.

4.2 Retrieval Provider Comparison

A controlled comparison between retrieval providers was conducted in order to evaluate how retrieval quality affects downstream fact-checking performance. The comparison focused primarily on Serper, Tavily, and Exa under progressively refined retrieval settings.

The provider comparison was performed while keeping the verifier model and evaluation subsets fixed as much as possible. This makes the observed differences more directly attributable to retrieval behavior rather than to changes in the verifier itself.

Serper was mainly used during earlier experiments and served as an initial retrieval baseline. Although functional, it frequently returned noisy snippets and weaker metadata, making evidence filtering and temporal validation more difficult.

Tavily produced more stable retrieval behavior and became the primary provider during the development of the heuristic retrieval pipeline. However, publication-date metadata was often unavailable or incomplete, limiting the effectiveness of leakage-safe temporal filtering.

The provider comparison did not show a single provider dominating every setting. Tavily was stronger in clean retrieval mode, while Exa slightly outperformed Tavily in heuristic mode and returned substantially more publication-date metadata. For this reason, the Exa heuristic run was selected as the final heuristic web baseline before the proposed-system comparison.

The difference between Tavily and Exa was small in absolute performance terms. Tavily clean retrieval reached 0.411 accuracy and 0.407 Macro F1, while Exa clean retrieval reached 0.393 accuracy and 0.390 Macro F1. In heuristic mode, however, Exa slightly led Tavily, with 0.486 accuracy and 0.479 Macro F1 compared with 0.477 accuracy and 0.473 Macro F1. Exa also provided substantially better publication-date metadata in the recorded runs, which made it more useful for the leakage-safe CheckIT transfer setting. The provider choice was therefore pragmatic rather than absolute: Exa was selected because it was the best heuristic run under the fixed pipeline and because its metadata better supported the thesis evaluation protocol.

The provider comparison showed that retrieval quality and metadata availability can substantially affect downstream verification performance, even when the verifier model and evidence-construction pipeline remain unchanged.

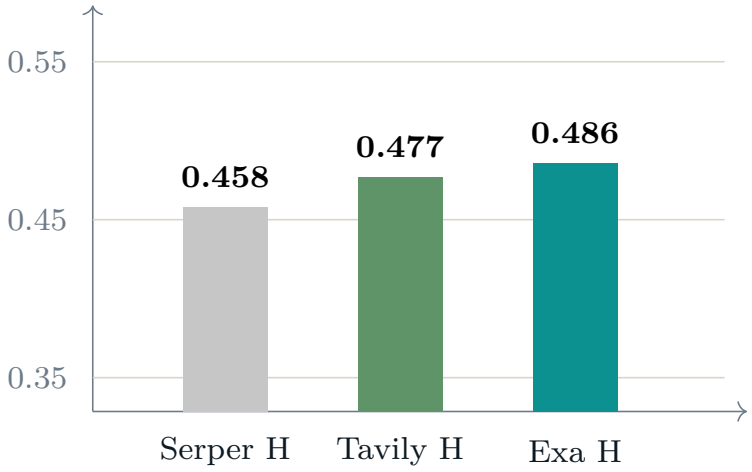

Figure 4.3: Provider comparison in heuristic mode on the FEVER-IT zero-shot error subset. Exa slightly led the heuristic setting and was selected for the final baseline and proposed-system runs.

4.3 CheckIT Evaluation

After the controlled FEVER-IT experiments, the evaluation was extended to CheckIT in order to analyze retrieval-based verification under more realistic political fact-checking conditions.

Compared to FEVER-IT, CheckIT introduces several additional challenges:

- higher retrieval leakage risk,
- stronger temporal dependence,
- more contextual political claims,
- and more ambiguous factual relations.

The CheckIT experiments therefore focus primarily on leakage-safe retrieval evaluation rather than on retrieval without these controls.

The experiments were conducted on the first 200 examples of the selected CheckIT subset. As in the FEVER-IT setting, the zero-shot baseline was evaluated first, followed by retrieval-based repair experiments performed only on the zero-shot error subset.

This evaluation design makes it possible to analyze whether retrieval augmentation can successfully repair claims that the verifier initially failed to classify correctly without external evidence. The zero-shot run produced 67 correct predictions and 133 errors, so the retrieval and proposed-system repair experiments were evaluated on those 133 error cases.

The same subset-size caveat applies here as in the FEVER-IT evaluation: the CheckIT results are controlled first-200-example results. This subset was chosen to keep leakage-safe web retrieval feasible while allowing consistent comparison between zero-shot, web-repair, and proposed-system settings.

4.3.1 Zero-Shot Results

The initial CheckIT experiment evaluates the zero-shot verifier without retrieval augmentation or external evidence.

The evaluation was conducted on the first 200 examples of the selected CheckIT subset using the same verifier configuration employed in the FEVER-IT experiments.

The zero-shot baseline achieved:

- Accuracy: 0.335
- Macro F1: 0.274

Compared to FEVER-IT, the performance drop is substantial. This reflects the higher difficulty of political fact-checking claims, which frequently depend on temporal context, institutional knowledge, implicit references, or numerical details that are not reliably captured by the verifier without retrieval.

The label distribution of CheckIT also contributes to the difficulty of the task. In particular, the dataset contains a strong skew toward REFUTES claims, while many political statements additionally involve nuanced factual relations that are harder to verify using only claim-level reasoning.

These results reinforce the importance of retrieval augmentation in realistic political fact-checking scenarios.

4.3.2 Leakage-Safe Retrieval Results

After the zero-shot evaluation, retrieval-based repair experiments were conducted on the resulting CheckIT error subset using leakage-safe retrieval constraints.

Before verification, the leakage-safe retrieval pipeline applied three constraints:

- domain filtering,
- temporal filtering,
- and evidence-construction heuristics.

The filtering logs show that these constraints had a substantial effect on the CheckIT web-repair run. Table 4.1 reports the main rejection counts for the 133 zero-shot error cases.

Table 4.1: Leakage-safe filtering statistics for the CheckIT Exa web-repair run.

Logged event	Count	Interpretation
Date-based rejections	535	Retrieved pages rejected because they were published on or after the claim date
Blocked-domain rejections	135	Candidate pages rejected because they came from known fact-checking domains
Fetch errors	13	Candidate pages that could not be fetched or parsed
Examples with zero usable URLs	24	Claims for which filtering and fetching left no usable evidence pages

These counts confirm that the protocol was not only a nominal restriction. It removed many pages that could have acted as answer pages, including direct professional fact-checking sources and pages published after the claim date. The goal is to verify claims using independent evidence, not by rediscovering the original fact-checking article.

The strongest leakage-safe heuristic retrieval configuration achieved:

- Repair Accuracy: 0.398
- Macro F1: 0.361

This corresponds to repairing 53 of the 133 CheckIT zero-shot errors.

The corresponding combined score over the original 200-example subset reached:

- Accuracy: 0.600
- Macro F1: 0.517

The leakage-safe setting is deliberately strict because many high-ranking pages containing professional fact-checking answers are removed during filtering.

Manual inspection of the outputs showed that the remaining retrieval failures were often caused by:

- indirect evidence,
- weak metadata,

- missing publication dates,
- or political claims requiring broader contextual interpretation.

Despite these limitations, the leakage-safe retrieval pipeline still repaired a substantial portion of the original zero-shot failures, suggesting that retrieval augmentation remains effective even under stricter evaluation constraints.

4.3.3 Proposed System Results

The final CheckIT experiments evaluate the proposed system under the same leakage-safe retrieval constraints used for the heuristic retrieval baseline.

The proposed system used the same verifier, filtering pipeline, and retrieval provider as the strongest heuristic configuration, while adding iterative query refinement and evidence sufficiency evaluation through the LangGraph workflow.

The strongest leakage-safe proposed-system configuration achieved:

- Repair Accuracy: 0.451
- Macro F1: 0.370

This corresponds to repairing 60 of the 133 CheckIT zero-shot errors.

The corresponding combined score over the original 200-example subset reached:

- Accuracy: 0.635
- Macro F1: 0.518

The combined CheckIT view is useful because repair scores are measured only on the 133 zero-shot errors, while the combined score shows the end-to-end effect after keeping the 67 zero-shot correct predictions.

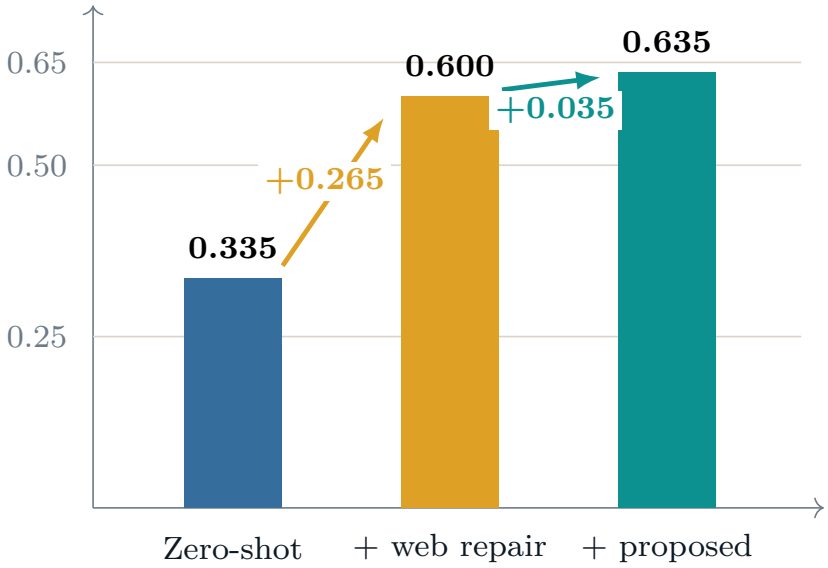

Figure 4.4: CheckIT combined accuracy. The combined score keeps zero-shot correct predictions and adds the claims repaired by leakage-safe retrieval or by the proposed system.

Compared to the heuristic retrieval baseline, the improvements remain modest but clearer than in FEVER-IT. The proposed system repaired 7 more cases than the web-repair baseline, raising combined accuracy from 0.600 to 0.635. However, combined Macro F1 changed only from 0.517 to 0.518, so the result should be interpreted as a small descriptive improvement rather than as strong evidence of a general agentic advantage. The largest gains were observed on claims where the first retrieval attempt returned only partial or weakly connected evidence, and where query refinement was able to recover more specific factual information.

The proposed workflow did not refine every claim. In the 133 CheckIT repair cases, 88 completed after the initial retrieval attempt, while 45 entered at least one query-refinement step. Of these, 37 reached the second refinement iteration. The give-up path returned NOTENOUGHINFO in 13 cases when the evidence remained too weak after the allowed iterations. These counts show that the proposed system was used as a controlled retrieval extension rather than as an unconstrained browsing agent.

At the same time, several difficult claims remained unresolved even after iterative retrieval. In many of these cases, the available web evidence was incomplete, highly contextual, or strongly dependent on political interpretation rather than on directly verifiable factual statements.

The CheckIT experiments therefore reinforce a pattern already observed on FEVER-IT: retrieval quality and evidence construction remain the dominant

factors in verification performance, while the proposed system provides smaller but measurable improvements once a strong heuristic baseline is already in place.

4.4 Additional Model Checks

After the main `gemma3:latest` experiments, additional model checks were run to test whether the CheckIT difficulty was specific to the main verifier model or whether it appeared across nearby open models. These checks used `llama3.1:8b`, `mistral:7b`, and `qwen2.5:7b`.

The additional runs were not intended to replace the main controlled comparison. Instead, they provide a robustness check for the interpretation of the CheckIT results. Each model was first evaluated in zero-shot mode on the first 200 examples of FEVER-IT and CheckIT. Then, for CheckIT, Exa web repair was applied to each model's own zero-shot error subset and recombined with that model's zero-shot correct predictions.

Table 4.2: Additional model checks on FEVER-IT and CheckIT.

Model	FEVER Acc.	FEVER F1	CheckIT Acc.	CheckIT F1	CheckIT + web Acc.	CheckIT + web F1
`gemma3:latest`	0.465	0.462	0.335	0.274	0.600	0.517
`llama3.1:8b`	0.520	0.491	0.300	0.254	0.580	0.496
`mistral:7b`	0.510	0.500	0.140	0.154	0.530	0.480
`qwen2.5:7b`	0.475	0.478	0.085	0.072	0.505	0.418

The FEVER-IT zero-shot results show that other models can match or exceed `gemma3:latest` in the more controlled setting. On CheckIT, however, none of the additional models produced a stronger zero-shot result or a stronger combined web-repair result than `gemma3:latest`. This suggests that the CheckIT difficulty is not only a weakness of the main verifier model. It is also related to the dataset, the political domain, and the evidence-retrieval setting.

The web-repair results also show that retrieval helps all tested models on CheckIT, including models with weak zero-shot performance. For example, `mistral:7b` and `qwen2.5:7b` start from very low zero-shot accuracy on CheckIT, but their combined scores improve substantially after web repair. However, their final combined scores remain below the main `gemma3:latest` pipeline.

The very low CheckIT zero-shot score for `qwen2.5:7b` should be treated as an outlier in this prompt and label-mapping setting rather than as a general statement about the model family. The run is still useful because it shows that the retrieval-repair protocol can improve a weak claim-only starting point, but the main comparison remains the controlled `gemma3:latest` pipeline.

A smaller proposed-system check was also run with `llama3.1:8b` on its CheckIT zero-shot error subset. This run reached 0.345 repair accuracy and 0.258 Macro F1

on 139 errors, corresponding to 0.545 combined accuracy and 0.456 combined Macro F1 over the original 200 examples. This remained below both the `llama3.1:8b` web-repair setting and the final `gemma3:latest` proposed-system result. Because this was not a complete sweep over all models, it is treated as a supporting check rather than as a central comparison.

4.5 Error Analysis

Manual inspection of the system outputs revealed several recurring failure modes across both FEVER-IT and CheckIT. Although retrieval augmentation improved overall verification performance, many difficult claims remained challenging even after heuristic filtering and iterative retrieval refinement.

One of the most common issues was false SUPPORTS predictions. In these cases, the retrieved evidence was strongly related to the topic of the claim but did not directly establish the factual relation required for verification. Partial entity overlap or thematic similarity often caused the verifier to produce overconfident SUPPORTS predictions despite missing decisive evidence.

The class-level repair-subset reports in Appendix A.4 support this observation. On the FEVER-IT error subset, the Exa heuristic baseline predicted SUPPORTS 66 times even though only 23 examples had SUPPORTS as the gold label. The proposed system reduced this to 60 SUPPORTS predictions and slightly improved NOTENOUGHINFO and REFUTES behavior, but many false SUPPORTS cases persisted. On CheckIT, the proposed system improved REFUTES recall on the repair subset, but NOTENOUGHINFO precision remained very low because many cases predicted as NOTENOUGHINFO did not belong to that class.

Another recurring problem involved numerical and temporal reasoning. Claims containing dates, quantities, percentages, or comparative statistics frequently produced incorrect predictions when retrieved evidence contained partially matching numerical information or temporally inconsistent details.

The experiments also exposed weaknesses in contradiction detection. Some retrieved documents indirectly contradicted the claim without containing explicit contradiction phrases, making the verifier more likely to miss the inconsistency.

In CheckIT, several failures were additionally caused by political and contextual ambiguity. Certain claims depended heavily on institutional context, historical interpretation, or implicit assumptions that were difficult to recover from web retrieval alone.

The proposed system partially reduced some retrieval failures by refining weak initial queries and recovering more targeted evidence. However, even iterative retrieval often remained limited by incomplete web evidence, weak metadata, or insufficient factual specificity in the retrieved documents.

Overall, the error analysis points to retrieval quality and evidence specificity as the main bottlenecks in realistic retrieval-based fact-checking systems.

Error category	Observed pattern
False SUPPORTS	The retrieved evidence is topically related but does not prove the factual relation in the claim.
Numbers and dates	Partially matching statistics, dates, or quantities lead the verifier toward the wrong conclusion.
Implicit contradictions	Contradictory evidence is indirect and can be missed when the verifier focuses on overlapping entities.
Political context	Some claims depend on institutional or historical background that is not directly retrieved.

Figure 4.5: Recurring error categories observed during manual inspection. The main remaining failures are connected to evidence specificity, numbers and dates, implicit contradictions, and political context.

4.6 Final Results Summary

Table 4.3 summarizes the main quantitative results reported in this chapter. The table is intended as a compact view of the full experimental progression: claim-only verification, oracle or web evidence, heuristic retrieval, and the proposed controlled retrieval workflow.

Table 4.3: Final summary of the main thesis results.

System	Dataset	Evaluation view	Accuracy	Macro F1
Zero-shot	FEVER-IT	200 examples	0.465	0.462
Oracle evidence	FEVER-IT	200 examples	0.625	0.589
Exa heuristic	FEVER-IT	107 zero-shot errors	0.486	0.479
Proposed system	FEVER-IT	107 zero-shot errors	0.495	0.488
Zero-shot	CheckIT	200 examples	0.335	0.274
Zero-shot + web repair	CheckIT	200 combined	0.600	0.517
Zero-shot + proposed system	CheckIT	200 combined	0.635	0.518

The FEVER-IT repair rows report performance on the 107 examples that the zero-shot verifier initially classified incorrectly. The CheckIT combined rows instead keep the zero-shot correct predictions and add the errors repaired by web retrieval or by the proposed system. Read together, the results show that retrieval produces the largest improvement over claim-only verification, while the proposed workflow adds a smaller measured gain once strong retrieval baselines are already in place.

4.7 Chapter Summary

This chapter evaluated retrieval-based systems and the proposed controlled retrieval workflow across controlled and realistic Italian fact-checking benchmarks.

The FEVER-IT experiments established the main comparison framework between zero-shot verification, oracle verification, plain retrieval, heuristic retrieval, and the proposed workflow. The results showed that retrieval can repair a substantial portion of zero-shot errors, while controlled query refinement provides smaller gains once strong heuristic retrieval baselines are already in place.

The CheckIT experiments further demonstrated the difficulty of realistic political fact-checking evaluation under leakage-safe retrieval constraints. Although retrieval augmentation remained effective, many failures persisted because of contextual ambiguity, indirect evidence, and temporal complexity.

The provider comparison additionally showed that retrieval quality strongly affects downstream verification performance, particularly in retrieval-constrained settings. The additional model checks supported the same interpretation: CheckIT remains difficult across several open models, while retrieval repair improves all tested models in combined evaluation.

The next chapter discusses the broader implications of these results, the limitations of the current approach, and possible future directions for retrieval-based fact-checking systems.

Chapter 5

Discussion and Conclusion

5.1 Discussion

The experiments show that retrieval augmentation is useful, but only when retrieval produces evidence that is specific enough for the verifier to use. The central result is therefore retrieval-first: evidence quality and evidence construction explain more of the observed performance change than workflow complexity alone. The proposed system is useful as a controlled extension to retrieval, but its gains are small compared with the effect of moving from claim-only verification to evidence-grounded verification.

5.1.1 Answer to RQ1

RQ1 asked how well an LLM can verify claims without evidence. The answer is that claim-only verification provides a useful baseline, but it is not reliable enough for realistic fact-checking. On FEVER-IT, `gemma3:latest` reached 0.465 accuracy and 0.462 Macro F1 on the first 200 examples. This shows that the model can verify some stable or well-known claims from parametric knowledge. On CheckIT, however, zero-shot accuracy fell to 0.335 and Macro F1 to 0.274. The drop indicates that political claims are more dependent on context, timing, and specific evidence than the controlled FEVER-IT setting.

The zero-shot results also show why claim-only evaluation can be misleading. A correct zero-shot answer may come from reasoning, from memorized knowledge, or from dataset exposure during training. An incorrect answer, by contrast, identifies a case where external evidence has a clear opportunity to help. This motivated the zero-shot error-repair protocol used throughout the thesis.

5.1.2 Answer to RQ2

RQ2 asked whether web retrieval can repair zero-shot errors. The experiments support a positive but qualified answer. On the FEVER-IT zero-shot error subset, Exa heuristic retrieval repaired 52 of 107 errors, reaching 0.486 accuracy and 0.479 Macro F1. On CheckIT, leakage-safe Exa web repair corrected 53 of 133 zero-shot errors. In combined form, the CheckIT score improved from 0.335 zero-shot accuracy to 0.600 with web repair.

These gains show that many claim-only failures are evidence failures: the verifier needs access to relevant web evidence before it can make a better decision. At the same time, retrieval is not enough by itself. Oracle evidence on FEVER-IT raised accuracy to 0.625 but still left many errors, especially for REFUTES and implicit contradictions. This means that even perfect or near-perfect evidence does not remove all verifier and label-interpretation difficulties.

The proposed system adds only a modest extension to this retrieval effect. On FEVER-IT, it repaired 53 errors instead of 52, a difference of one case on the 107-example repair subset. On CheckIT, it repaired 60 errors instead of 53 and improved combined accuracy from 0.600 to 0.635, but combined Macro F1 changed only from 0.517 to 0.518. The proposed system is therefore best interpreted as a small measured improvement from controlled query refinement, not as a decisive gain from agentic orchestration.

5.1.3 Answer to RQ3

RQ3 asked whether the protocol works on realistic political claims. The CheckIT experiments show that it works in the sense that it can be applied under stricter leakage-aware constraints and still improves over claim-only verification. The filtering logs are important here: leakage-safe web repair rejected 535 pages because of publication dates, 135 pages because of blocked fact-checking domains, and produced 24 examples with no usable URL. These counts show that the evaluation was not simply retrieving answer pages.

At the same time, CheckIT remains difficult. Political claims often require institutional context, exact dates, numerical comparisons, and interpretation of public statements. The additional model checks show that this difficulty is not specific to `gemma3:latest`: other open models performed similarly or worse after combined web repair. The protocol therefore transfers to political fact-checking, but the results also reveal the limits of current retrieval-based verification in this setting.

5.1.4 Broader Interpretation

The main theoretical implication is that retrieval-augmented fact-checking should not be evaluated only as a final classification problem. It should also be evaluated as an evidence-grounding problem. Many remaining errors were not failures to output a valid label; they were failures to obtain or identify decisive evidence. This was clearest in false SUPPORTS cases, where pages were topically related to the claim but did not prove the specific relation being checked.

This also changes how the proposed system should be understood. The agentic component is not the main contribution by itself. Its value is conditional: it helps when query refinement retrieves more specific evidence or when the judge prevents verification on weak evidence. When retrieval remains generic, noisy, or temporally unreliable, additional workflow control cannot compensate for the missing evidence.

5.2 Limitations

The most important limitation is the size of the controlled evaluation subsets. The main retrieval comparisons use 107 FEVER-IT zero-shot errors and 133 CheckIT zero-shot errors. These subsets were appropriate for controlled web-retrieval experimentation, runtime constraints, and manual inspection, but they limit statistical strength. Small differences, especially the 52-versus-53 repaired-case result on FEVER-IT, should be treated as descriptive and not statistically significant. No McNemar test, bootstrap confidence interval, or equivalent uncertainty analysis was performed.

A second limitation concerns class-level analysis. The appendix reports selected per-label results for the main runs, but the thesis does not provide a complete per-class breakdown for every provider, model, and system variant. Macro F1 helps reduce the effect of label imbalance, but it can still hide which labels drive a result. This matters because false SUPPORTS predictions and NOTENOUGHINFO calibration were recurring failure modes.

The heuristic retrieval components were also not ablated individually. The thesis reports the combined heuristic pipeline because that was the strongest non-agentic baseline, but it does not quantify how often each guard or ranking signal fired. As a result, the experiments show the value of the heuristic pipeline as a whole rather than the isolated effect of each heuristic.

Retrieval leakage is reduced but not eliminated. Domain blocking and date filtering remove many direct answer pages, but open-web retrieval can still surface indirect summaries, syndicated material, pages with missing dates, or content derived from the original fact-checking article. The leakage-safe protocol should therefore be interpreted as a stricter and more transparent approximation of realistic retrieval, not as a guarantee that all leakage has been removed.

The experiments are also affected by the variability of web retrieval. Search-provider rankings, publication-date metadata, page availability, and fetched page contents can change over time. This makes exact rerunning difficult even when the verifier model, prompts, and parameters are fixed.

The retrieval and proposed-system parameters were fixed as practical defaults rather than tuned through a full hyperparameter search. Values such as `k_search`, `k_fetch`, evidence length, page length, and iteration count were chosen to support controlled comparison under resource constraints. Different defaults may change the balance between recall, precision, runtime, and verifier stability.

Evidence extraction remains noisy. BeautifulSoup-based preprocessing removes many HTML artifacts, but boilerplate, navigation text, tables, cookie banners, and encoding problems can still enter evidence blocks. This affects both retrieval quality and verifier behavior, especially when the decisive factual detail appears only in a small part of a long page.

Finally, the verifier itself remains limited. Numerical comparisons, temporal consistency, implicit contradictions, weak-evidence calibration, and REFUTES detection all produced errors even when relevant evidence was available. The choice of a local 4B verifier made the experiments practical and reproducible, but stronger or differently prompted models may change the absolute scores.

5.2.1 Threats to Validity

Internal validity. The main internal threats come from web-search variability, retrieval nondeterminism, provider ranking differences, and changing page availability. Even with a fixed verifier, fixed prompts, and fixed retrieval parameters, a web-based system can observe different evidence at different times.

External validity. The experiments use first-200 example subsets and focus on Italian fact-checking datasets. This makes the evaluation practical and comparable across systems, but it limits how far the absolute scores can be generalized to larger datasets, other languages, other political contexts, or different claim genres.

Construct validity. Repair accuracy measures whether a previous zero-shot error becomes correct after evidence is added. This is useful for isolating the value of retrieval, but it is not the same as full end-to-end fact-checking quality. Leakage-safe retrieval also reduces direct answer-page lookup but cannot eliminate leakage completely, and retrieval quality strongly affects the measured verification performance.

5.3 Future Work

Future work should first scale the evaluation to larger subsets and report statistical uncertainty. Confidence intervals, paired tests, and larger CheckIT samples would

make it easier to distinguish robust gains from noise, especially for small differences between heuristic retrieval and the proposed system.

More detailed label-level and error-category analysis is also needed. Future evaluations should report per-label precision, recall, and F1 for every main system, together with counts for false SUPPORTS, over-abstention, missed REFUTES, temporal errors, and numerical errors.

Retrieval quality remains the most important technical direction. Future systems could explore stronger ranking models, better evidence selection, source reliability scoring, temporal metadata extraction, and multi-document reasoning. These improvements would target the main bottleneck observed in this thesis: evidence that is related to the claim but not decisive.

Leakage control can also be improved. More robust publication-date extraction, dataset-specific source blocking, duplicate-page detection, and detection of mirrored fact-checking content would make political fact-checking evaluation stricter and more reproducible.

The proposed-system parameters should be tuned systematically in future work. Retrieval depth, fetch depth, evidence length, judge threshold, and maximum iterations may all affect the balance between useful query refinement and unnecessary extra retrieval.

Finally, memory and structured knowledge remain possible extensions. The current system keeps only short-term state during a single verification run. Future workflows could store previously verified claims, reuse trusted evidence, or combine web retrieval with structured knowledge bases and source reliability graphs.

5.4 Conclusion

This thesis investigated LLM-based fact-checking as an evidence-grounded retrieval problem. Across FEVER-IT and CheckIT, the strongest conclusion is that retrieval quality is the main driver of performance. Claim-only verification is useful as a baseline, but it is not enough for realistic political fact-checking.

Web retrieval repaired many zero-shot errors, while leakage-safe filtering made the CheckIT evaluation more realistic by reducing direct answer-page lookup. The proposed controlled retrieval workflow provided a small measured extension through query refinement and evidence sufficiency judging, but its gains were modest and should be interpreted cautiously.

The work therefore argues for a restrained view of agentic fact-checking. More workflow control can help, but only when it improves the evidence available to the verifier. For trustworthy automated fact-checking, the central challenge remains evidence grounding: retrieving independent, timely, specific, and usable evidence before a verification label is produced.

Taken together, the results leave three practical takeaways. First, retrieval substantially improves over claim-only verification, but the quality of the retrieved evidence matters more than the complexity of the surrounding workflow. Second, strong heuristic retrieval already captures much of the obtainable gain, while the proposed workflow adds a smaller measured improvement through controlled query refinement. Third, leakage-safe evaluation is necessary for realistic political fact-checking, because unrestricted retrieval can otherwise turn the task into answer-page lookup rather than independent verification.

Appendix A

Additional Experimental Tables

This appendix reports compact backup tables for the main experimental results discussed in Chapter 4. The tables are included to make the reported comparisons easier to inspect without interrupting the main discussion.

A.1 FEVER-IT Main Results

Table A.1: FEVER-IT main results used in the controlled evaluation.

System	Subset	Accuracy	Macro F1	Notes
Zero-shot	200 examples	0.465	0.462	Claim-only verifier
Oracle evidence	200 examples	0.625	0.589	Gold dataset evidence
Exa heuristic retrieval	107 zero-shot errors	0.486	0.479	52 repaired cases
Exa proposed system	107 zero-shot errors	0.495	0.488	53 repaired cases

A.2 Retrieval Provider Comparison

Table A.2: Provider comparison on the FEVER-IT zero-shot error subset.

Provider/run	Accuracy	Macro F1	Role in development
Serper clean	0.308	0.300	Early historical run
Serper heuristic	0.458	0.426	Early historical run
Tavily clean	0.411	0.407	Strong clean retrieval run
Tavily heuristic	0.477	0.473	Strong heuristic run
Exa clean	0.393	0.390	Stable clean retrieval run
Exa heuristic	0.486	0.479	Final heuristic web baseline

A.3 CheckIT Repair and Combined Results

Table A.3: CheckIT repair results and combined scores over the original 200 examples.

System	Subset	Accuracy	Macro F1	Notes
Zero-shot	200 examples	0.335	0.274	67 correct cases
Exa web repair	133 zero-shot errors	0.398	0.361	53 repaired cases
Exa proposed system	133 zero-shot errors	0.451	0.370	60 repaired cases
Zero-shot + web repair	200 combined	0.600	0.517	120 correct cases
Zero-shot + proposed system	200 combined	0.635	0.518	127 correct cases

The combined CheckIT accuracy is computed by keeping the zero-shot correct predictions and adding the number of repaired errors. For example, the web-repair configuration combines 67 already-correct zero-shot predictions with 53 repaired cases, giving 120 correct predictions out of 200.

A.4 Per-Label Breakdown for Main Runs

Table A.4 reports selected per-label precision, recall, and F1 values for the main runs. The table is included to make the Macro F1 scores easier to interpret. It is not a complete per-label analysis for every provider and model variant.

Table A.4: Selected per-label reports for the main FEVER-IT and CheckIT runs.

Dataset/run	Subset	Label	Precision	Recall	F1	Support
FEVER-IT zero-shot	200 examples	SUPPORTS	0.472	0.646	0.545	65
FEVER-IT zero-shot	200 examples	REFUTES	0.628	0.397	0.486	68
FEVER-IT zero-shot	200 examples	NOTENOUGHINFO	0.353	0.358	0.356	67
FEVER-IT oracle	200 examples	SUPPORTS	0.680	0.970	0.800	65
FEVER-IT oracle	200 examples	REFUTES	0.770	0.250	0.380	68
FEVER-IT oracle	200 examples	NOTENOUGHINFO	0.530	0.670	0.590	67
FEVER-IT Exa heuristic	107 errors	SUPPORTS	0.318	0.913	0.472	23
FEVER-IT Exa heuristic	107 errors	REFUTES	0.724	0.512	0.600	41
FEVER-IT Exa heuristic	107 errors	NOTENOUGHINFO	0.833	0.233	0.364	43
FEVER-IT proposed system	107 errors	SUPPORTS	0.333	0.870	0.482	23
FEVER-IT proposed system	107 errors	REFUTES	0.688	0.537	0.603	41
FEVER-IT proposed system	107 errors	NOTENOUGHINFO	0.733	0.256	0.379	43
CheckIT Exa web repair	133 errors	SUPPORTS	0.486	0.367	0.419	49
CheckIT Exa web repair	133 errors	REFUTES	0.667	0.395	0.496	76
CheckIT Exa web repair	133 errors	NOTENOUGHINFO	0.098	0.625	0.169	8
CheckIT proposed system	133 errors	SUPPORTS	0.469	0.306	0.370	49
CheckIT proposed system	133 errors	REFUTES	0.689	0.553	0.613	76
CheckIT proposed system	133 errors	NOTENOUGHINFO	0.075	0.375	0.125	8

The FEVER-IT repair results show that the heuristic and proposed-system runs still over-predict SUPPORTS relative to the gold distribution. The CheckIT repair results show a different pattern: the proposed system improves REFUTES recall on the error subset, while NOTENOUGHINFO remains difficult because only 8 repair-subset examples have this gold label and many non-NOTENOUGHINFO claims are predicted as NOTENOUGHINFO. The very low NOTENOUGHINFO precision for the CheckIT proposed-system run should therefore be read as evidence of over-abstention on a small class rather than as a stable standalone estimate of class performance.

A.5 Additional Model Checks

Table A.5: Additional model checks on FEVER-IT and CheckIT. CheckIT web scores are combined over each model's original 200-example subset.

Model	FEVER Acc.	FEVER F1	CheckIT Acc.	CheckIT F1	CheckIT + web Acc.	CheckIT + web F1
`gemma3:latest`	0.465	0.462	0.335	0.274	0.600	0.517
`llama3.1:8b`	0.520	0.491	0.300	0.254	0.580	0.496
`mistral:7b`	0.510	0.500	0.140	0.154	0.530	0.480
`qwen2.5:7b`	0.475	0.478	0.085	0.072	0.505	0.418

Table A.6: Additional `llama3.1:8b` proposed-system check on CheckIT.

System	Subset	Accuracy	Macro F1	Notes
`llama3.1:8b` + Exa proposed system	139 zero-shot errors	0.345	0.258	Repair subset
`llama3.1:8b` + Exa proposed system	200 combined	0.545	0.456	Combined score

A.6 Fixed Retrieval Parameters

Table A.7: Fixed retrieval and proposed-system parameters used in the final experiments.

Parameter	Value	Purpose
Search provider	Exa	Final provider after the FEVER-IT provider comparison
Main verifier	`gemma3:latest` / `gemma3:4b`	Local 4B verifier kept fixed for the main controlled comparison
Temperature	0.0	Reduces unnecessary verifier variation
`k_search`	8	Retrieves enough candidates without making each query too expensive
`k_fetch`	4	Fetches a manageable number of pages for evidence construction
`page_max_chars`	12000	Prevents very long pages from dominating the evidence block
`evidence_max_chars`	5000	Keeps the final evidence block usable for the local verifier
`max_iterations`	2	Allows query refinement while limiting runtime and instability

A.7 Prompt Templates

This section summarizes the prompt templates used by the verifier and the proposed workflow. The exact prompt wording is stored in the implementation used to produce the JSONL outputs. The implementation wraps these templates in the local LangChain/Ollama calling code, but the behavioral contract is the same across runs: the model must return a constrained JSON object and must choose only from SUPPORTS, REFUTES, and NOTENOUGHINFO.

Table A.8: Prompt templates used in the verification pipeline.

Prompt	Input	Required output	Main instructions
Zero-shot verifier	Claim text	JSON with `label` and `reasoning`	Classify the claim as SUPPORTS, REFUTES, or NOTENOUGHINFO without retrieved evidence; use the label set exactly
Evidence-based verifier	Claim plus evidence block	JSON with `label`, `reasoning`, and `sources`	Use only the provided evidence; return NOTENOUGHINFO when evidence is incomplete or does not address the claim directly
Heuristic verifier variant	Claim plus ranked evidence block	JSON with `label`, `reasoning`, and `sources`	Apply the same evidence-only rule, with stronger instructions for contradiction cues, negations, exclusivity, numbers, and dates
Planner	Claim text	JSON with `query`	Produce one short factual search query suitable for web retrieval
Judge	Claim plus current evidence block	JSON with `enough_evidence` and `reason`	Return true only when evidence directly supports or contradicts the decisive claim relation; return false for generic, partial, or weak evidence
Query refinement	Claim, current query, and judge reason	JSON with `query`	Produce a more specific query that targets the missing factual anchor identified by the judge

The evidence-based verifier prompt follows the rule: use only the evidence block, never unsupported model knowledge. The judge prompt uses a stricter sufficiency criterion: evidence is sufficient only when it addresses the decisive relation in the claim, including crucial names, dates, numbers, roles, and relations. When the judge returns `false`, the refinement prompt uses the judge's reason to form a more targeted query for the next search iteration.

Bibliography

[1] Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. «A Survey on Automated Fact-Checking». In: *Transactions of the Association for Computational Linguistics* 10 (2022), pp. 178–206. DOI: 10.1162/tacl_a_00454 (cit. on pp. 1, 5–9, 11).

[2] Jacopo Gili, Lucia Passaro, and Tommaso Caselli. «CheckIT!: A Corpus of Expert Fact-checked Claims for Italian». In: *Proceedings of the Ninth Italian Conference on Computational Linguistics (CLiC-it 2023)*. Venice, Italy, 2023. URL: https://ceur-ws.org/Vol-3596/paper27.pdf (cit. on pp. 1, 5, 9, 10, 13).

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. «Attention Is All You Need». In: *Advances in Neural Information Processing Systems*. Vol. 30. 2017. URL: https://arxiv.org/abs/1706.03762 (cit. on pp. 1, 6).

[4] Tom B. Brown et al. «Language Models are Few-Shot Learners». In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020, pp. 1877–1901. URL: https://arxiv.org/abs/2005.14165 (cit. on pp. 1, 6).

[5] Long Ouyang et al. «Training Language Models to Follow Instructions with Human Feedback». In: *Advances in Neural Information Processing Systems*. Vol. 35. 2022, pp. 27730–27744. URL: https://arxiv.org/abs/2203.02155 (cit. on pp. 1, 6).

[6] 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*. Association for Computational Linguistics, 2018, pp. 809–819. DOI: 10.18653/v1/N18-1074. URL: https://arxiv.org/abs/1803.05355 (cit. on pp. 5, 8, 9, 11).

[7] Reiichiro Nakano et al. «WebGPT: Browser-assisted Question-answering with Human Feedback». In: *arXiv preprint arXiv:2112.09332* (2021). URL: https://arxiv.org/abs/2112.09332 (cit. on pp. 6–8, 11).

[8] Patrick Lewis et al. «Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks». In: *Advances in Neural Information Processing Systems*. Vol. 33. 2020, pp. 9459–9474. URL: https://arxiv.org/abs/2005.11401 (cit. on pp. 6, 7, 11).

[9] 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 International Conference on Learning Representations*. 2023. URL: https://arxiv.org/abs/2210.03629 (cit. on pp. 7, 8, 18).

[10] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. «Toolformer: Language Models Can Teach Themselves to Use Tools». In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023. URL: https://arxiv.org/abs/2302.04761 (cit. on pp. 7, 8, 18).

[11] Antonio Scaiella, Stefano Costanzo, Elisa Passone, Danilo Croce, and Giorgio Gambosi. «Leveraging Large Language Models for Fact Verification in Italian». In: *Proceedings of the Tenth Italian Conference on Computational Linguistics (CLiC-it 2024)*. Pisa, Italy, 2024. URL: https://ceur-ws.org/Vol-3878/97_main_long.pdf (cit. on p. 9).

[12] Yucheng Li, Frank Guerin, and Chenghua Lin. «An Open Source Data Contamination Report for Large Language Models». In: *arXiv preprint arXiv:2310.17589* (2023). URL: https://arxiv.org/abs/2310.17589 (cit. on p. 13).

[13] Cheng Xu, Shuhao Guan, Derek Greene, and M.-Tahar Kechadi. «Benchmark Data Contamination of Large Language Models: A Survey». In: *arXiv preprint arXiv:2406.04244* (2024). URL: https://arxiv.org/abs/2406.04244 (cit. on p. 13).

[14] Gemma Team et al. «Gemma 3 Technical Report». In: *arXiv preprint arXiv:2503.19786* (2025). URL: https://arxiv.org/abs/2503.19786 (cit. on p. 21).