

Learning to Fact Check

Towards a general Framework for Multimodal Fact-Checking

Master thesis in the department of Computer Science by Marcus Kornmann

Date of submission: August 28, 2026

1. Review: Prof. Dr. Anna Rohrbach
2. Review: Mark Rothermel, M. Sc.
Darmstadt



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Computer Science
Department
Multimodal AI Lab

Erklärung zur Abschlussarbeit gemäß § 22 Abs. 7 APB TU Darmstadt

Hiermit erkläre ich, Marcus Kornmann, dass ich die vorliegende Arbeit gemäß § 22 Abs. 7 APB der TU Darmstadt selbstständig, ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Ich habe mit Ausnahme der zitierten Literatur und anderer in der Arbeit genannter Quellen keine fremden Hilfsmittel benutzt. Die von mir bei der Anfertigung dieser wissenschaftlichen Arbeit wörtlich oder inhaltlich benutzte Literatur und alle anderen Quellen habe ich im Text deutlich gekennzeichnet und gesondert aufgeführt. Dies gilt auch für Quellen oder Hilfsmittel aus dem Internet.

Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, August 28, 2026

Marcus Kornmann

Abstract

Automated fact-checkers apply one verification procedure to every claim, whether it is a written assertion, an authentic photograph placed in a false context, or a manipulated video. This thesis asks whether choosing that procedure per claim improves accuracy, and it builds a representation in which the question can be measured. Each verification procedure, called a *blueprint*, names the checks a class of claim requires, the actions that settle them, and the conditions under which it should be selected. A pool of blueprints is induced from published fact-checks, with no human annotation of the investigative process. An LLM reduces each training article to a topic-stripped fingerprint, those are clustered, and one blueprint is synthesized per cluster. At inference a router assigns each claim a single blueprint, which guides a multi-agent pipeline of a planner, a web-search agent, a media agent that checks the provenance of attached images and video, and a judge.

Blueprints learned from 2025 fact-checks were applied to the 1,000 claims of 2026 Q1, under two planner models and five judges. Procedures survive the induction, and claim text alone predicts which one a fact-checker followed at 83.9% on a held-out quarter. Conditioning verification on that choice did not improve accuracy. It helped the weaker planner and hurt the stronger one, and the routed pool never reached what the stronger planner achieves unguided. Taking that negative apart locates the binding constraint. Judging on the evidence a human fact-checker gathered cuts error by roughly 85%, so the verdict stage is close to saturated. Retrieval recovers under half of those findings, and the pipeline never attempts four in five of the sources the fact-checkers used. The error lives upstream of everything the strategy layer controls, in the stage that decides which sources to consult.

Blueprints learned from fact-checks published in 2025 were applied to the 1,000 VERITAS claims of the first quarter of 2026, under two planner models and five judges. The induction yields a pool of distinct procedures, and claim text alone predicts which one a fact-checker followed at 83.9% on a held-out quarter. Conditioning verification on that choice did not improve accuracy. It helped the weaker planner and hurt the stronger one, and the routed pool never reached what the stronger planner achieves unguided. Taking that negative result apart locates the bottleneck, and it lies upstream of everything the strategy layer controls. Given the evidence a human fact-checker gathered, the judge cuts error by roughly 85%, so the verdict stage is close to saturated. Accuracy is lost earlier, in retrieval, which recovers under half of those findings and never attempts four in five of the sources the fact-checkers used. The method that shows this is cheap and general. With adequate evidence, a modern LLM judge already reaches the right verdict, so the verdict stage is close to solved and the accuracy still to be won lies in retrieval. Reporting evidence recall alongside the end-to-end score keeps that distinction legible, and separates a system that reasoned badly from one that was never shown the evidence to reason over.

Contents

1. Introduction	7
1.1. Approach	9
1.2. Contributions	9
1.3. Outline	10
2. Background	11
2.1. Reasoning in LLMs	11
2.1.1. Inference-time Scaling	12
2.1.2. Reinforcement Learning (RL) and Supervised Fine-Tuning (SFT)	13
2.1.3. Distillation	14
2.1.4. Limitations of Reasoning Models	15
2.2. Tool Use in LLMs	17
2.2.1. Training-Free Tool Use	18
2.2.2. Trained Tool Use	19
2.2.3. Limitations of Tool Use	20
2.3. Agentic Systems	21
2.3.1. Single-Agent Systems	22
2.3.2. Multi-Agent Systems	23
3. Related Work	25
3.1. Automated Fact-Checking	25
3.1.1. Task Scope	26
3.1.2. Modalities	29
3.1.3. Evidence Sources	30
3.1.4. Verdicts and Justifications	32
3.1.5. Benchmarks	32
3.2. Strategy Learning	33
3.2.1. Reflection and Episodic Memory	34
3.2.2. Automatic Prompt and Instruction Optimization	34
3.2.3. Automated Agentic Workflow and System Design	35
3.2.4. Tool and Library Induction	35
3.2.5. Unsupervised Structure Discovery	36
3.3. Research Gap	36
4. Methods	38
4.1. Blueprints	38
4.2. Blueprint Selection	41
4.2.1. Claim Features	41

4.2.2.	Stage One: Rule-Based Filtering	43
4.2.3.	Stage Two, Mechanism A: LLM Tie-Break	43
4.2.4.	Stage Two, Mechanism B: Embedding Probe	43
4.2.5.	Combining the Two Mechanisms	44
4.3.	Agent-Based Blueprint Execution	44
4.3.1.	Agent Framework	44
4.3.2.	Fact-Check Agent	45
4.3.3.	Web-Search Agent	48
4.3.4.	Media Agent	50
4.3.5.	Judge Agent	52
4.4.	Blueprint Learning	53
4.4.1.	Notation	53
4.4.2.	Strategy Fingerprints	54
4.4.3.	Clustering	55
4.4.4.	Per-Cluster Synthesis	55
4.4.5.	Consolidation	56
4.4.6.	Mechanical Guards	57
4.4.7.	Routing Probe	58
4.5.	Single-File Strategy Baseline	58
4.5.1.	Construction: Sequential Distillation	59
4.5.2.	Inference	60
5.	Experiments	61
5.1.	Experiment Setup	61
5.1.1.	Data	61
5.1.2.	Models	62
5.1.3.	The Learned Blueprint Pool	64
5.1.4.	Compared Systems	64
5.1.5.	Pipeline configuration	67
5.1.6.	Metrics	67
5.1.7.	Evaluation protocol	68
5.2.	Main Results	68
5.2.1.	Comparison to the VERITAS baseline	71
5.2.2.	Qualitative Analysis	74
5.3.	Where the Error Comes From	76
5.3.1.	What Routing Contributes	76
5.3.2.	Judging on Ground-Truth Evidence	77
5.3.3.	What Retrieval Failure Costs	79
5.4.	Ablations	82
5.4.1.	Refining with unused Budget	82
5.4.2.	Does Manipulation Detection Help?	83
6.	Discussion and Future Work	86
6.1.	Answers to the Research Questions	86
6.2.	Why the Blueprint Effect Reverses	87
6.3.	What the Remaining Error Is Worth	88
6.4.	Why Procedural Guidance Misses Sourcing	89



6.5. What Each Metric Needs Beside It	89
6.6. Limitations	90
6.7. Future Work	91
7. Conclusion	93
A. Appendix	112
A.1. Configuration Studies	112
A.1.1. Judge Configuration	112
A.1.2. Evidence Budget and Presentation	113
A.1.3. Blueprints	113
A.2. Cost and Runtime	114
A.3. Tool Analyses	114
A.3.1. Geo-Location Tool Analysis	114
A.3.2. Manipulation Tool Analysis	115
A.4. Calibrating the Judge	118
A.5. Qualitative Sample	119
A.6. Confusion Matrices	121
A.7. The Learned Blueprint Pool	128
A.8. The Learned Strategy Document	145

1. Introduction

Misinformation reaches large audiences before it can be corrected, and false claims travel faster and further online than the truths they crowd out [135]. That is the problem fact-checking exists to address, and it works. In a large-scale study of community-based fact-checking on X, exposure to a community note reduced the subsequent spread of a misleading post by 61.2% on average, and raised the odds that the author deleted the post altogether [27]. What limits the effect of this work is throughput. In interviews across 19 countries, 21 professional fact-checkers described work that is largely manual and spread across disconnected tools [90]. Participants in the same study reported being inundated with material that has to be filtered and prioritized before verification can begin. Ethnographic accounts of the practice show why the work resists compression, since establishing what a claim actually asserts and what evidence would settle it is itself an interpretive act [43]. Every claim that reaches a fact-checking organization competes for a fixed and small amount of expert attention, and the claims that lose that competition are simply never checked.

The volume of material competing for this attention keeps growing, and its character is changing with it. Generative models have lowered the cost of producing plausible falsehoods and have made them harder to recognize once produced. Chen and Shu [21] show that misinformation rewritten by a large language model, with its semantics preserved and only its style altered, is more difficult to flag for human readers and automated detectors alike. A multi-year survey [34] of media-based misinformation in the wild traces how quickly generative tools entered the material that fact-checkers actually handle. Synthetic text has spread through open platforms at a pace that is easy to underestimate. On Medium, the share of posts attributed to a language model climbed from statistical noise to more than a third in under three years, and Quora followed the same curve [124]. Reddit, over the same window, barely moved. The shift is concentrated in platforms publishing standalone articles and answers, which sit closer to what a verification system will treat as a source than a comment thread does.

Claims also arrive in more than one modality. Misinformation carried by images or video is judged more credible than its text-only counterpart, and it travels faster [4]. The dominant pattern is recontextualization rather than fabrication, where an authentic photograph is paired with a caption that misstates when, where, or why it was taken [132, 82, 1]. Benchmarks have been slow to follow. Most multimodal fact-checking datasets cover one or two languages and a single task [38], while the claims reaching fact-checkers are neither monolingual nor confined to one form of manipulation. VERITAS [110] was constructed to reflect these conditions. This is one of the reasons why it is used as the benchmark in this work. Of the 1000 claims in the quarter this thesis evaluates on, 849 carry at least one image or video. They span 41 languages and 88 fact-checking organizations, led by English, Spanish, and Hindi.

Automated fact-checking has been pursued as a research problem for close to a decade [130, 48], and current systems assemble large language models into pipelines that retrieve evidence, reason over it, and return a verdict together with an explanation [136]. Evaluation is beginning to move off static benchmarks. Li and Bakker [73] report a three-month study on X in which an LLM pipeline wrote 1614 contextual notes on live

posts through the platform’s note-writing program, measured against human-written notes on the same posts. The setting was live, though the system still operated under researcher supervision.

In the multimodal setting, DEFAME [16] is representative of the current state of the art. It plans its own tool calls over a toolkit of web search, reverse image search, and geolocation, and returns to planning for a further round until it commits to a verdict. Accuracy nonetheless falls well short of what unsupervised deployment would require. On the first-quarter 2026 split of VERITAS, its strongest configuration reaches 58.1% accuracy on the seven-point integrity scale and 88.3% once predictions are collapsed to the three-way distinction between  **Intact**,  **Unknown**, and  **Compromised**. The same model, given a single web-search tool, a smaller query budget, and one pass over the evidence, records 0.281 mean squared error against DEFAME’s 0.282, while trailing it by six points of seven-class accuracy. Two things follow. Verdicts are still wrong on roughly one claim in eight at the coarsest granularity available, and the iterative multi-tool pipeline earns its far larger retrieval and inference budget in exact-label accuracy alone. Both observations shape the questions this thesis asks.

Research on automated fact-checking has largely proceeded by decomposition. Separate lines of work address claim matching [115], evidence retrieval and provenance [25], out-of-context image detection [1, 103], manipulation localization [47], and explanation generation [10], each against its own benchmark [48, 4]. End-to-end systems assemble several of these capabilities into one pipeline. None of them varies the procedure itself. The control flow a claim passes through is fixed by the architecture, so a fabricated quotation, a photograph recycled from an earlier event, and a disputed mortality statistic are all handled the same way.

Professional practice looks different. Fact-checkers converge on a verdict through investigative routines that differ by what kind of claim they are facing, and ethnographic accounts describe the choice of approach as part of the expertise rather than as a preliminary to it [43, 90]. To put numbers on this, we ran the article analysis of Chapter 4 over the 4,000 fact-checks of 2025, recording for each the claim type, the investigative steps taken, and the evidence that settled the case. Both distributions are wide. No claim type accounts for even a third of the workload, from media authenticity (32.8%) down to identity claims (3.3%), and no single action settles more than 41% of claims, with source lookup, official records, expert consultation, and reverse image search each carrying a substantial share. The workload is procedurally diverse, and a single fixed control flow cannot match it.

That distribution motivates a straightforward hypothesis. If expert verification is procedurally diverse and if the appropriate procedure is at least partly predictable from the claim itself, then a system that selects among procedures should outperform one that applies a single procedure to everything. Testing this requires committing to three things that prior work leaves open: (1) a representation in which a verification procedure can be written down and executed, (2) a mechanism for acquiring such procedures at scale rather than authoring them by hand, and (3) a rule for deciding which one a given claim should receive. No existing system does this. Adaptive behavior in current pipelines operates on tool selection within a fixed control flow, and no published work learns a repertoire of verification procedures from data or routes claims among them.

This thesis addresses three questions.

1. **Can verification strategies be learned from published fact-checks?** Fact-checking organizations document their reasoning in the articles they publish. We ask whether that record is sufficient to recover reusable procedures automatically, without expert annotation of the procedures themselves.
2. **Does conditioning verification on a per-claim strategy improve accuracy?** Given a pool of learned strategies and a routing mechanism, we ask whether the resulting system outperforms an otherwise identical pipeline that runs one strategy for every claim.

-
3. **If it does not, where does the error actually come from?** A negative answer to the second question is informative only if it is accompanied by an account of what the binding constraint is instead. We therefore decompose the remaining error and attribute it to the stage of the pipeline that produces it.

1.1. Approach

We build a multi-agent pipeline that takes a claim, optionally with attached media, and returns an integrity rating on the seven-point VERITAS scale [110]. A planner decides what to investigate and when to stop, a web-search agent retrieves and extracts evidence, a media agent checks the provenance and location of attached images and video, and a judge assigns the final label. Every run is recorded, so a verdict can be traced to the stage that produced it.

To condition verification on the claim, the system needs a procedure it can name and reuse per claim type. That object is a *blueprint*, a declarative specification naming the checks a verdict requires, the actions that settle them, and the conditions under which it should be selected. The planner receives it as guidance rather than as a program, so the blueprint fixes what is investigated while the planner keeps its freedom over how.

Blueprints are learned rather than authored. An LLM reduces each training article to a topic-stripped *strategy fingerprint* of claim type, decisive actions, and evidence sequence; fingerprints are clustered, one blueprint is synthesized per cluster, and a consolidation pass merges near-duplicates. At inference a router assigns each claim exactly one blueprint.

Two baselines hold every other setting fixed and give every claim a single playbook, one distilled from the same articles and one empty. Blueprints learned from fact-checks published in 2025 are evaluated on the 1,000 VERITAS claims of the first quarter of 2026, under two planners and five judges. We then decompose the remaining error and attribute it to the stage that produces it.

1.2. Contributions

This thesis contributes the following:

- **A blueprint-based multimodal fact-checking system.** Verification procedures are represented as data rather than control flow and executed over multimodal claims, and the procedure a claim needs is learnable from its text alone. The system matches or beats the published VERITAS baseline and commits to a wrong verdict direction less often than any baseline.
- **A method for learning verification procedures from published fact-checks.** The published record alone induces a pool of executable procedures, with no manual annotation of the investigative process and no hand curation, together with the guards that make a generated pool internally consistent.
- **Evidence that per-claim strategy conditioning does not improve accuracy.** The effect reverses with planner strength, helping the weaker planner and hurting the stronger, with all five judges agreeing, and a single distilled playbook recovers most of the gain, which places the benefit in the guidance and not its routing. One mechanism explains both halves. An unresolved check buys a further retrieval pass where the missing evidence is reachable, and drags the verdict toward the middle of the scale where it is not.

-
- **A diagnosis of the binding constraint.** On the evidence a human fact-checker gathered, the judge cuts error by roughly 85% and misjudges direction on under 1% of claims, so the verdict stage is close to saturated. Retrieval recovers under half of those findings, pipeline error tracks that recall while claim difficulty does not, and four in five of the sources the fact-checkers used are never attempted.
 - **Component findings that stand independently.** General-purpose multimodal models beat specialized geolocation models by roughly thirty points of top-1 country accuracy, on public and never-published images alike. Forensic manipulation detection does not survive redistribution, and even an oracle detector leaves the verdict unchanged. Verdict certainty is not recoverable from trace features, and roughly four fifths of the apparent gain from a refinement stage is resampling rather than new evidence.

1.3. Outline

Chapter 2 covers the capabilities an automated fact-checker is assembled from, namely reasoning, tool use, and agentic control. Chapter 3 surveys the fact-checking systems built out of those capabilities and the benchmarks they report on, turns to the literature on learning reusable procedures for LLM agents, and states the gap this thesis addresses. Chapter 4 describes the blueprint representation, the routing stage, the agents that execute a blueprint, the procedure that learns a pool of them from published fact-checks, and the single-document baseline the pool is compared against. Chapter 5 specifies the experimental setup, reports the main comparison, attributes the error that remains, and closes with two component ablations. Chapter 6 discusses what follows from those results, states the limitations of the work, and names what it leaves open. Chapter 7 concludes the thesis. The appendix reproduces the learned blueprint pool and the strategy document as the pipeline loaded them and reports the tool analyses and configuration studies behind the settings used throughout. All code is published on GitHub¹ and visualizations of traces produced by the system, as well as of the blueprints, can be inspected in a viewer².

¹<https://github.com/CptK/mafc-blueprint-learning>

²<https://cptk.github.io/mafc-blueprint-learning/index.html> (Note: images are not available here)

2. Background

Fact-checking a claim requires three things: deciding what evidence would settle it, obtaining that evidence, and inferring a verdict from what comes back [4]. Human checkers proceed this way, tracing a claim to its origin and triangulating sources that are individually inconclusive [43]. An automated system inherits the same structure, and each demand corresponds to one capability. Inferring the verdict from material that rarely addresses the claim head-on is a *reasoning* problem. Obtaining the evidence is a *tool-use* problem, because the evidence lies outside the model. Claims worth checking usually postdate the training corpus, and audiovisual claims call for operations such as reverse image search and geolocation. Deciding the steps is a problem of *orchestration*, because neither their number nor their order can be fixed in advance. Whether a step such as reverse image search is worth running depends on what earlier queries returned as well as other factors. Verification therefore unfolds as a loop of planning, acting, and revising.

These three capabilities structure the chapter. Reasoning is treated in Section 2.1, tool use in Section 2.2, and the agentic systems that supply the orchestration in Section 2.3. One question runs through all three, namely where a system’s procedure comes from and whether it transfers between tasks. A capability can be obtained in several ways, whether elicited by a prompt, trained into the weights, transferred from a larger model, or accumulated across episodes. These routes place procedural knowledge in different locations, and they differ in whether experience gained on one task survives into the next. This thesis puts that question to fact-checking, so the chapter attends throughout to where a system’s procedure originates.

Each section closes with the limitations that recur for its capability, and two bear directly on verification. A trace that rationalizes a conclusion, rather than reporting how it was reached, undermines the justification a fact-check is expected to deliver. Adversarial content in retrieved material attacks the step that supplies the grounding [180]. Concrete fact-checking systems and benchmarks follow in Chapter 3.

2.1. Reasoning in LLMs

Fact-checking requires inferring a verdict from evidence that seldom settles the claim on its own, which is the reasoning demand set out in the chapter introduction. This section defines what reasoning means for an LLM and reviews how the capability is elicited, trained, and transferred.

Johnson-Laird’s mental model theory describes human reasoning as the construction of *mental models*, internal representations of what could be true given a set of premises [65]. Reasoning then proceeds by exploring these models rather than by manipulating symbols according to formal rules. A conclusion is judged valid if it holds across all models consistent with the premises. Systematic errors arise when people consider too few of them, for instance by overlooking a counterexample. Two points carry over to what follows. First, reasoning is better understood as the construction and exploration of intermediate representations of a problem than as a single-step mapping from premises to conclusion. Second, the reliability of that process depends on how many such representations are considered along the way.

In LLMs, reasoning is likewise characterized as the ability to draw inferences from premises or evidence [12]. What distinguishes reasoning-oriented models is the route to those inferences. They produce intermediate steps and re-evaluate prior actions and responses instead of mapping an input directly to a final answer. Most current reasoning-capable LLMs make this explicit by emitting a dedicated reasoning or "thinking" trace, separated from the final answer.

Inference itself takes several forms. *Deduction* draws a conclusion that must hold whenever its premises hold. *Induction* generalizes from particular observations to a rule they support but do not guarantee. *Abduction* infers the hypothesis that best explains an incomplete set of observations [64, 166]. Fact-checking relies mainly on the latter two, because evidence seldom entails a verdict outright and the checker must decide which conclusion individually inconclusive sources best support. In natural language processing, this judgment is formalized as *natural language inference* (NLI), the task of labeling the relation between a premise and a hypothesis as entailment, contradiction, or neutrality [15, 31]. A verdict of supported, refuted, or not enough information corresponds to these three labels, applied to a claim and its retrieved evidence.

The remainder of this section examines three approaches to obtaining such reasoning behavior in LLMs: (1) eliciting it at inference time through prompting and search strategies, (2) training it directly through supervised fine-tuning and reinforcement learning, and (3) transferring it from a larger model into a smaller one through distillation. Finally, it discusses limitations that persist across these approaches.

2.1.1. Inference-time Scaling

Inference-time scaling is the simplest way of achieving reasoning in LLMs without modifying the model itself, instead allocating additional compute at inference time. Some variants pair this with a separately trained verifier or revision model [28, 121], but the reasoning behavior still comes from how compute is spent at inference rather than from how the model was trained. Wei et al. introduced CHAIN-OF-THOUGHT (CoT) prompting [147], where the model is given manually written exemplars that include the intermediate derivation leading to each answer, so that it produces its own derivation before the final answer. They showed that reasoning traces in the prompt change what the model produces, and that the effect is emergent rather than gradual. In the models they studied, CoT yielded little benefit below roughly 100B parameters, where chains were fluent but invalid, while above that threshold gains grew with the complexity of the problem. That threshold reflects prompting alone, and the training (Section 2.1.2) and distillation (Section 2.1.3) methods discussed below elicit reasoning at far smaller scales. Kojima et al. reduce CoT to its simplest trigger and show that hand-crafted exemplars are not needed at all [68]. Appending a single fixed phrase ("Let's think step by step") elicits a reasoning trace in a zero-shot setting, which a second prompt then converts into a parseable final answer. Zero-shot CoT remains scale-dependent and still trails carefully engineered few-shot CoT, but it closes much of the gap with one template reused across tasks.

Sampling-based approaches predate CoT and scale compute by breadth rather than by length. Cobbe et al. generate multiple full solutions and select among them with a verifier trained to judge correctness [28], recovering much of the gain that would otherwise require a substantially larger generator. The benefit is not monotonic in the number of samples, since accuracy eventually degrades as adversarial candidates begin to fool the verifier. SELF-CONSISTENCY [141] combines this sampling with CoT. Rather than generating one reasoning trace, the model samples several independent CoT traces for the same problem and takes the answer that appears most often, using majority vote in place of a learned verifier. This needs no additional training, but applies only where outputs map onto a fixed answer set. TREE OF THOUGHTS [163] generalizes parallel sampling into a search. It decomposes a problem into intermediate reasoning steps that form a branching tree, and the model evaluates partial solutions at each node. Unpromising branches are pruned by backtracking,

rather than committing to a fixed set of independently sampled solutions. Yao et al. note that plain prompting, CoT, self-consistency, and self-refinement all fall out as special cases, obtained by restricting the tree and the search procedure.

Orthogonal to parallel sampling, SELF-REFINE [85] iteratively improves a single solution. The model critiques its own output and revises it based on that self-generated feedback, repeating until the solution is judged sufficient. Snell et al. show that the best strategy depends strongly on problem difficulty [121]. Easy questions benefit more from sequential refinement of a single answer, while harder ones benefit more from parallel search across many candidates. Allocating compute adaptively on this basis is several times more compute-efficient than a best-of- N baseline, and on easy to intermediate questions a smaller model given additional inference compute can outperform a far larger one. Zhou et al. show that such adaptation does not require an explicit difficulty estimate. Their ADAPTIVE-SOLVER framework [177] judges the reliability of the current answer by consistency across samples, and escalates along a pipeline of increasingly expensive prompting methods, sample sizes, and models only when that check fails.

2.1.2. Reinforcement Learning (RL) and Supervised Fine-Tuning (SFT)

One prominent instantiation of RL for LLMs is Reinforcement Learning from Human Feedback (RLHF) [95]. The pipeline starts with supervised fine-tuning (SFT), where a pretrained model is trained on human-written demonstrations of the desired behavior. The SFT model’s outputs are then ranked by human annotators, and these rankings train a separate reward model. The LLM is optimized against that reward model using Proximal Policy Optimization (PPO) [114], with a KL penalty that keeps it close to the SFT checkpoint. The effect is large enough that a 1.3B RLHF-tuned model is preferred by human raters over the 175B GPT-3 baseline. Optimizing against the reward model does cost general capability, which Ouyang et al. largely recover by mixing pretraining gradients back into the RL updates [95]. Group Relative Policy Optimization (GRPO) reduces the cost of this optimization step by dropping PPO’s learned value model, instead sampling a group of responses per prompt and estimating each one’s advantage from how far its reward deviates from the group mean [117]. A different line of simplification targets the reward model rather than the optimizer. Direct Preference Optimization (DPO) optimizes the model directly on preference data, reparameterizing the reward in terms of the policy and a frozen reference model so that the same KL-constrained objective becomes a classification loss over preferred and dispreferred responses [108]. This removes the reward model and the reinforcement learning loop alike.

DEEPSEEK-R1-ZERO departs from this lineage entirely. It is built on top of the DEEPSEEK-V3 base model and applies pure RL with GRPO, using no SFT and no learned reward model or preference data of any kind [32]. Reward comes from two rule-based signals. Accuracy rewards evaluate whether the model’s response is objectively correct. For math problems with deterministic answers, the model must provide the final answer in a specified format (e.g. boxed), enabling reliable rule-based verification. For coding competition problems, a compiler runs the generated code against predefined test cases to check correctness. Format rewards enforce that the model encapsulates its reasoning within designated `<think>` and `</think>` tags, kept separate from the final answer in `<answer>` tags, a separation the authors motivate as aiding interpretability. Learned reward models are deliberately avoided here, since these are prone to reward hacking during large-scale RL. This minimal setup is enough for reasoning behaviors to emerge without any demonstration data, among them self-reflection, verification, and adaptation of strategy part-way through a solution. Similar RL-driven emergence has since been reproduced by open efforts such as OPEN-REASONER-ZERO [61], which pares the recipe down further. It rewards only answer correctness, drops the format reward and KL regularization entirely, and returns to PPO with a learned critic on the grounds that the critic assigns credit at the token level and devalues

repetitive output. From the same 32B base model it surpasses the corresponding DEEPSEEK-R1-ZERO variant while using roughly a tenth of the training steps. Despite demonstrating that reasoning can emerge from RL alone, R1-Zero’s outputs suffer from poor readability and language mixing.

To address these shortcomings, the full DEEPSEEK-R1 adopts a multi-stage pipeline that alternates between SFT and RL, rather than relying on RL alone [32]. First, a small amount of curated long chain-of-thought data is used to fine-tune the base model, instilling a clean, readable `<think>` format before any RL takes place. Second, the resulting checkpoint is trained with the same rule-based accuracy and format rewards as R1-Zero, now supplemented with a language-consistency reward that stabilizes the output language at a small cost in raw accuracy. Third, the resulting model generates a large number of candidate responses, from which the best are selected via rejection sampling and mixed with general-purpose, non-reasoning data. This combined dataset of roughly 800K samples is used for a further round of SFT, restoring general capabilities that narrowed during the reasoning-focused RL stage, much as the pretraining mix does in RLHF. Finally, a second RL phase adds model-based preference rewards for helpfulness and safety alongside the rule-based reasoning rewards, refining the model across all task types rather than only verifiable ones. Reward hacking surfaces in precisely this phase. Reward scores kept rising while performance on CODEFORCES, a competitive-programming benchmark, fell, which bears out the earlier decision to keep learned reward models out of DEEPSEEK-R1-ZERO. A similar SFT-then-RL pattern for eliciting reasoning has been reported elsewhere, for instance in KIMI k1.5 [129], which likewise combines a long-CoT SFT warm-up with subsequent RL. It departs from DEEPSEEK-R1 in how correctness is established. Where R1 constrains the final answer into a fixed format so that a rule can verify it, KIMI k1.5 permits free-form answers and trains a reward model to judge whether a response matches the ground truth, since the same mathematical answer admits many written forms. The variant used during RL first generates a step-by-step assessment and then emits a verdict, which raised judging accuracy from roughly 84% for a conventional value-head reward model to 98.5% in the authors’ spot checks. The signal being learned is therefore still correctness rather than preference, and value functions and process reward models are avoided throughout.

2.1.3. Distillation

Classically, in distillation a smaller student model is trained on the softened output distribution of a larger teacher, obtained by raising the temperature of the teacher’s softmax, optionally combined with the ground-truth labels through a weighted sum of the two objectives [56]. These softened targets carry information that hard labels discard, namely the relative probabilities the teacher assigns to the remaining classes. With the rise of reasoning models, distillation has become relevant in a different sense, referring to supervised fine-tuning of smaller models on rationales or outputs generated by a larger teacher rather than transferring soft targets directly. The shift is partly a practical one. Soft targets require access to the teacher’s output distribution, whereas sampled reasoning traces can be obtained from any model that can be prompted, including one served behind an API.

DISTILLING STEP-BY-STEP [60] uses CoT prompting on a teacher LLM to extract both rationales and final answers, then trains a smaller model to predict the label and the rationale as two separate tasks. Treating the rationale as an additional output rather than an additional input is the key design choice, since conditioning on a rationale at inference time would keep the large model in the loop. The approach reduces both the model size needed to match a larger LLM’s performance and the amount of labeled training data required, compared to standard fine-tuning or distillation. A 770M T5 student outperforms the few-shot prompted 540B PALM teacher on one benchmark while using only 80% of the available data. Magister et al. [86] instead annotate an existing dataset with CoT rationales generated by a teacher model. Providing the gold answer to the teacher

during this generation step helps it correct errors in its own reasoning, and rationales that still fail to reach the correct answer are filtered out before training. The student is then trained to produce both the reasoning and the final answer given only the original question. Distilled versions of DEEPSEEK-R1 dispense with the annotation step altogether. Several smaller base models from the QWEN and LLAMA families, ranging from 1.5B to 70B parameters, are fine-tuned via SFT directly on the same approximately 800K curated samples used in DEEPSEEK-R1’s own third training stage, without any additional RL applied to the distilled models themselves [32]. This outperforms running reasoning RL on the small model directly. KIMI K1.5 [129] targets a related transfer, using long-CoT models to improve short-CoT ones and reporting better token efficiency from doing so with RL than with DPO, rejection sampling, or model merging. Distillation does not exceed its source, since the student inherits the teacher’s failure modes along with its reasoning patterns.

2.1.4. Limitations of Reasoning Models

While the previous sections outline how reasoning capabilities in LLMs are elicited, trained, and transferred, the resulting behavior falls short of what the term "reasoning" might suggest at face value. Four limitations recur across the approaches above. They concern whether a model’s reasoning trace can be trusted as a faithful account of its underlying computation, whether the reasoning is used efficiently, whether training incentivizes shortcuts in place of the intended task, and whether the apparent ability is as robust as benchmark performance suggests.

Faithfulness A common assumption is that a model’s CoT explanation reveals its actual underlying reasoning process, but this does not hold in general. Agarwal et al. [3] distinguish between the plausibility of an explanation, i.e. whether it appears logical and coherent to a human reader, and its faithfulness, i.e. whether it actually reflects the model’s underlying reasoning process. They argue that the field’s emphasis on user-friendly, conversational explanations has generally favored the former at the expense of the latter, and that LLMs are plausible by default, having been trained on human-written text and then tuned by RLHF to present answers in human-like form. Turpin et al. [133] demonstrate the gap concretely. Appending a phrase that suggests a particular multiple-choice option causes GPT-3.5 and CLAUDE 1.0 to adopt that option, with accuracy dropping by as much as 36% across thirteen BIG-BENCH HARD tasks, while the generated CoT never mentions the suggestion as a factor. In 73% of these cases the stated reasoning actively argues for the biased answer, and in 15% it contains no detectable error at all, so the trace is a coherent rationalization, not merely an incomplete one. These demonstrations rely on an injected cue, but unfaithfulness is not confined to manipulated prompts. Arcuschin et al. [9] find it in naturally phrased questions, where a model gives contradictory answers to a question and its reverse while defending each with coherent reasoning, a pattern they call implicit post-hoc rationalization. Lanham et al. [70] test faithfulness more directly by perturbing CoT traces, e.g., through truncation, introduced mistakes, or paraphrasing, and checking whether the final answer changes as a result. They find that faithfulness varies substantially across tasks, and that smaller models are, counterintuitively, often more faithful than larger ones on the same tasks. A control in the same study rules out one candidate explanation for CoT’s benefit. Replacing the trace with uninformative filler tokens produces no accuracy gain, so the benefit does not come from the additional test-time compute alone.

Chua and Evans [26] narrow the question to reasoning-trained models, testing whether a model’s CoT explicitly acknowledges when a prompt cue influenced its final answer. Reasoning models articulate such influential cues far more often than their non-reasoning counterparts, though the rate stays inconsistent across cue types. They also offer a partial explanation for the gap, since a GPT-4o judge used as a proxy reward model prefers the less faithful of two responses in 63% to 71% of comparisons, which suggests that preference-based

training discourages explicit articulation. Chen et al. [24] reach a similar conclusion using a comparable hinted-versus-unhinted multiple-choice setup on CLAUDE 3.7 SONNET and DEEPSEEK-R1. Across six hint types, reasoning models are more faithful than non-reasoning ones, but overall faithfulness remains low, and it degrades further on harder questions. Outcome-based RL raises faithfulness early in training and then plateaus well short of reliability. Most consequentially for what follows, in synthetic environments where models learn to exploit a reward hack on over 99% of examples, they verbalize that hack in fewer than 2%, which suggests that CoT-based monitoring scales poorly to exactly the behaviors it would most need to catch. A more basic question follows. Faithfulness presupposes that the model reasons in the language it emits, yet work on reasoning in a continuous latent space treats the verbal trace as one projection of a computation that need not be verbal [52]. On that view, some gap between trace and computation is expected rather than a defect to be trained away.

Overthinking and Inefficient Reasoning Beyond faithfulness, a second limitation concerns the efficiency of the reasoning process itself, commonly referred to as *overthinking*, where reasoning models generate excessively long chains of thought that do not yield corresponding gains in accuracy [123]. Chen et al. [23] show that on math problems a correct answer is typically already present in the first of several solution attempts within a single response, in over 92% of cases, while later parts of the response repeat the same reasoning strategy rather than exploring new ones. The additional reasoning is therefore largely redundant, particularly on simpler tasks. Cuadron et al. [29] extend this analysis to agentic settings, where reasoning models must interleave reasoning with actions in an environment, such as software engineering tasks. They define overthinking there as an excessive reliance on internal reasoning at the expense of environmental feedback, and identify three recurring failure patterns: analysis paralysis, rogue actions, and premature disengagement. Higher overthinking scores correlate strongly with lower task success, and reasoning models overthink more on average than non-reasoning models on the same tasks. Wang et al. [142] describe the mirror-image failure, which they term *underthinking*. Rather than repeating one strategy, models frequently abandon a promising line of reasoning before following it to completion, oscillating between strategies without committing to any one despite producing long chains of thought. More than 70% of their incorrect responses contain at least one correct thought that was dropped before it could be carried through.

Reward Hacking A third concern is that the training procedures described above can incentivize shortcuts rather than reasoning. During RL training, models can learn to exploit weaknesses in the reward signal itself instead of solving the intended task, a phenomenon known as reward hacking. Helff et al. [55] show that models trained with RL from verifiable rewards (RLVR) develop systematic shortcuts against a rule-based verifier on inductive reasoning tasks. Asked to induce a general rule, the models instead enumerate instance-level labels, which satisfies a verifier that only checks whether the output fits the examples it was given. Two observations indicate that this is a product of RLVR rather than of limited ability: non-RLVR models produce no such shortcuts on identical tasks, and the behavior disappears once the verifier additionally requires invariance under logically equivalent renamings of the problem. Shortcut prevalence also rises with task complexity and with inference-time compute, so allocating more reasoning does not make the behavior less likely. Baker et al. [11] observe related behavior in agentic coding, where models manipulate unit tests or terminate programs early to obtain a passing score without implementing a correct solution. Their central finding concerns oversight rather than the exploits themselves. A monitor that reads the reasoning trace detects such hacks far more reliably than one observing only actions and outputs, but once that monitor is folded into the training reward, models keep hacking while concealing their intent and monitor recall collapses. The authors therefore argue for accepting a monitorability tax, declining to optimize the trace directly in order to keep it informative. Nishimura-Gasparian et al. [94] situate these results within the broader pattern of

specification gaming and measure how training affects it across eight environments spanning coding, tool use, and agentic tasks. Every frontier model tested exploits its specification at non-negligible rates, RL reasoning training raises exploit rates by between 32% and 170% relative across matched model pairs, and test-time mitigations reduce the behavior without eliminating it. That such shortcuts are not confined to controlled benchmarks became apparent in July 2026. OpenAI reported that several of its models, evaluated on a cyber-capabilities benchmark with refusal safeguards deliberately disabled, chained a zero-day vulnerability in a package registry proxy with privilege escalation to reach the open internet. They then compromised Hugging Face’s production infrastructure to retrieve the evaluation’s solutions.¹

Brittleness to Superficial Variation A related but distinct limitation concerns the robustness of reasoning once training is complete. Models that solve a given problem often fail on superficial variations of the same problem. Mirzadeh et al. [91] generate many instantiations of the same grade-school math problem from symbolic templates and find that accuracy varies substantially across them. Substituting numerical values lowers performance modestly and widens that variance, while inserting a clause that appears relevant but contributes nothing to the solution causes drops of up to roughly 65 points for some models. Their stronger interpretive claim, that this reflects replication of training patterns rather than logical reasoning, has been contested, but the variance result alone is enough to make a single benchmark score an unreliable summary of reasoning ability. More recent evaluations confirm that the underlying brittleness is unresolved. Hou et al. [59] replace hand-written perturbation templates with an automated rewrite-and-verify loop that searches for semantically equivalent variants a specific model answers incorrectly, inducing failure rates above 60% on GSM8K and MATH-500 for most of the smaller models tested, although fine-tuning on the discovered variants recovers part of the gap. Hao et al. [53] show that frontier reasoning models are affected as well, evaluating them on mathematically equivalent transformations of competition-level problems. Renaming variables costs OpenAI’s O3 model 4.7 percentage points and resampling the numeric parameters while preserving the proof structure costs 12.9, an ordering that holds across capable models and indicates structural perturbations are harder than lexical ones. Taken together, these findings suggest that strong performance on standard benchmarks does not necessarily reflect robust, generalizable reasoning. They also carry a limitation of their own, since nearly all of this evidence comes from mathematics, code, and synthetic logic, leaving open how these failure modes manifest in natural-language verification tasks such as fact-checking.

These limitations are largely properties of reasoning considered in isolation, independent of how a model interacts with its environment.

2.2. Tool Use in LLMs

Reasoning is rarely deployed on its own. Models increasingly interleave it with the ability to invoke external tools, retrieve information, or act on an environment, which introduces both new capabilities and new failure modes. Whatever route produces the reasoning, an LLM remains constrained by what its parameters store and by what fits in its context. Parametric knowledge is fixed at training time, and since frequent retraining is impractical, models have no access to information that postdates training or falls outside their training distribution [89, 150]. Their computation is likewise approximate rather than executed. Arithmetic, symbolic manipulation, and other procedures with a uniquely correct answer are produced via learned pattern

¹<https://openai.com/index/hugging-face-model-evaluation-security-incident/>. The investigation was ongoing at the time of writing, with a third-party assessment by METR and Redwood Research pending, so these are preliminary findings reported by OpenAI.

completion rather than guaranteed-correct execution. A third gap is grounding. Models cannot observe the current state of a webpage, file, or external system, nor verify their own outputs against a source of truth outside their parameters. Even a reasoning process that decomposes a problem correctly is thus limited by what the model already knows and can compute internally. Equipping LLMs with external tools (for retrieval, code execution, or interaction with other models and systems) addresses these gaps and has become a central paradigm for extending what LLMs can reliably do beyond their parameters alone.

2.2.1. Training-Free Tool Use

Tool use can be obtained at inference time without updating the underlying LM's weights for any particular tool. MRKL [67] frames tool use as a routing problem. A router directs each incoming query to the module best suited to answer it, whether a symbolic expert such as a calculator, currency converter, or database API, or a smaller specialized LM, with the general-purpose LM serving as a fallback when no expert matches. The backbone LM is never retrained; only the router is, which the authors describe as a comparatively lightweight task. REACT [162] instead interleaves the decision into the reasoning trace itself. The model alternates between a "thought" that decomposes the task or interprets the latest result and an "action" such as a search or lookup, whose result returns as an "observation" and feeds the next thought. This grounds otherwise ungrounded reasoning in retrieved information, though it depends on task-specific exemplars. LATS [176] extends this loop with mechanisms seen earlier in this chapter, now applied to acting rather than pure reasoning. It embeds MONTE CARLO TREE SEARCH over candidate action sequences into REACT, scoring nodes with an LM-generated value function combined with self-consistency. Like TREE OF THOUGHTS, it uses the LM as its own evaluator. Unsuccessful trajectories trigger a self-reflection that summarizes the errors and proposes alternatives, in the manner of REFLEXION [120]. The result unifies reasoning, acting, and planning, but demands substantially more inference-time compute than REACT and assumes an environment whose earlier states can be restored. CRITIC [41] also closes the loop between generation and external verification, but makes verification an explicit, tool-interactive step. The model first produces an answer from parametric knowledge alone. It then interacts with a tool designated for the task, such as a search engine, a code interpreter, or a toxicity API, to check properties of that answer such as truthfulness or safety, and revises based on the critique this surfaces. The verify-then-correct cycle repeats until the critique confirms the answer or an iteration budget is exhausted. Unlike REACT's per-step grounding or LATS's search over action sequences, correction applies post-hoc to a complete draft rather than interleaved with its construction.

A parallel line of work targets LLMs' inability to execute exact computation rather than their lack of external information. PAL [37] and PROGRAM OF THOUGHTS [22] independently propose having the LLM generate a runnable program as its intermediate reasoning step. An external interpreter then carries out the arithmetic, indexing, or iteration, leaving the model responsible only for decomposition and program generation, never for computation itself. VIPERGPT [125] carries the same idea into visual queries, prompting a code model to compose vision and language APIs, such as object detection or image captioning, into a Python program that an interpreter then runs. HUGGINGGPT [119] treats entire expert models, rather than single-purpose APIs, as the callable tools. An LLM plans a sequence of subtasks, selects an appropriate expert model for each from a pool of available models, executes them, and synthesizes their heterogeneous outputs into one response. Across all these approaches, tool use is obtained without fine-tuning the underlying LM for the tools involved, whether by prompting it directly or, in MRKL's case, by training only a lightweight router around a frozen backbone.

2.2.2. Trained Tool Use

Tool use can also be learned directly, rather than elicited through prompting alone. TOOLFORMER [112] bootstraps this ability itself. Starting from a handful of hand-written examples per tool, the model generates candidate API calls throughout a text corpus, executes them, and keeps only those that measurably reduce its own loss on the subsequent text, then fine-tunes on the resulting self-annotated dataset. This teaches the model when and how to call a small fixed set of tools (a calculator, a question-answering system, Wikipedia search, machine translation, and a calendar) with no human labeling of tool use and no degradation of its general language modeling ability.

Scaling this idea to the much larger and less stable space of real-world APIs takes more than self-supervision over a handful of tools. TOOLLLM [105] builds an instruction-tuning dataset from real RapidAPI APIs and fine-tunes LLAMA-2 on it. Because the full API pool cannot fit in a prompt, a separately trained dense retriever recommends candidate APIs per instruction. The training trajectories themselves are annotated by CHATGPT under a depth-first search over a tree of candidate reasoning-and-action steps. At each node the annotator either proceeds along a promising path or abandons it and expands an alternative, and only successfully completed paths are retained. The same search doubles as an inference-time procedure, raising ChatGPT's pass rate under REACT. GORILLA [99] targets a related failure mode at a smaller scale. When APIs are numerous, overlapping, and subject to change, models hallucinate calls or get their syntax wrong. GORILLA fine-tunes on self-instructed instruction-API pairs drawn from three model hubs, and its retriever-aware training lets it condition on API documentation retrieved at inference time, so the model tracks documentation changes without being retrained.

Reasoning-style RL training, introduced in Section 2.1.2 for reasoning traces, has since been applied to tool use directly. RETOOL [36] follows the same cold-start-then-RL pattern as DEEPSEEK-R1. It first fine-tunes on code-augmented reasoning traces to instill the ability to interleave natural language with executable code. A second stage then applies RL, so the model learns when an interpreter call actually helps rather than reproducing the call patterns of the cold-start data. AVATAR [149] leaves the model's weights untouched and optimizes the instructions supplied to it instead. A comparator LLM samples queries from a training split, groups them into well-performing and poorly-performing sets, and contrasts the two to locate systematic failures in task decomposition, tool selection, and result synthesis. It then rewrites the instructions appended to the acting agent's prompt, repeating this offline over successive batches. The product is an optimized prompt rather than an updated model, which places AVATAR closer to training than to prompting despite computing no gradients, since it still requires labeled training queries.

In practice, current frontier models combine both paradigms, and the boundary between them falls within the individual tool. Models are trained to decide reliably when and how to invoke a tool. For tools the developer defines, the name, description, and argument schema are supplied only at inference time, typically as JSON schema definitions passed alongside the prompt (as in the OpenAI² or Anthropic³ APIs). The MODEL CONTEXT PROTOCOL (MCP) standardizes this interface across providers, exposing external tools and data sources to a model through a common schema rather than a vendor-specific one.⁴ Providers also ship tools whose schemas are fixed and trained on, such as a bash shell or a text editor. This function-calling capability generalizes the small fixed tool sets of TOOLFORMER or GORILLA to tools the model has never seen during training, though it makes performance contingent on how well each tool is described in its schema.

²<https://developers.openai.com/api/docs/guides/function-calling>

³<https://platform.claude.com/docs/en/agents-and-tools/tool-use/overview>

⁴<https://modelcontextprotocol.io>

2.2.3. Limitations of Tool Use

While the previous sections outlined how tool-invoking behavior in LLMs is elicited through prompting and instilled through training, the resulting behavior falls short of the reliable, general-purpose tool use the term might suggest. The following limitations concern whether a model selects and invokes the correct tool in the first place, whether it strikes an appropriate balance between internal reasoning and interacting with its environment, and whether the content a tool returns can itself be trusted as a reliable basis for the model's subsequent behavior.

Tool Selection and Invocation Errors API-BANK [75] establishes tool-call correctness as a distinct evaluation axis for LLM agents. Its benchmark comprises manually annotated multi-turn dialogues over a runnable system of real APIs, graded at three levels of difficulty according to whether the relevant tool is known in advance, must first be retrieved, or must be retrieved as part of a multi-step plan. Accuracy degrades sharply once retrieval enters the picture, and the error analysis shows that models fail in characteristically different ways. Weaker instruction-tuned models frequently issue no call at all. GPT-4 fails predominantly at retrieving the right API and at parsing outputs that contain several calls. The small model the authors fine-tune on their own training set, LYNX, most often hallucinates API names outright, a pattern that recurs below. Xiong et al. [154] examine parameter-filling failures in more depth, treating correct tool-agent trajectories as reference solutions and systematically perturbing the information available at each stage of the invocation pipeline, namely the user query, the tool documentation, and previously returned tool outputs. The perturbations map onto distinct failure categories. Stripping parameters out of the user query is the strongest driver of task deviation, where the agent pursues something other than what was asked. Corrupting type information in the tool documentation drives specification mismatch instead, and malformed tool returns produce missing-information failures, though less severely than either of the other two sources. Hallucinated parameter names behave differently. They stay flat across every perturbation the authors apply, which locates that failure mode in the model rather than in noisy or incomplete input. The study covers single-turn, API-style invocations in English, so the taxonomy's reach beyond that setting is untested. Across both studies, the same failure mode proves intrinsic to the model rather than to its input, namely the hallucination of tool or parameter names, flat under every input perturbation and most frequent in the smallest models. Its consequences reach into how training data is collected. ToolLLM's depth-first search with backtracking (Section 2.2.2) exists because single-path prompting traps even GPT-4 in faulty loops of incorrect and hallucinated calls, often failing to find a valid solution path at all [105].

Over-Reliance and Under-Reliance on Tools Beyond errors in individual calls, the overthinking patterns of Section 2.1.4 are, read from the tool side, three distinct ways of misjudging when to act. Analysis paralysis and premature disengagement are both failures to invoke a tool at all, the first before acting and the second before verifying. Rogue actions is the inverse failure of not consuming what a tool returned, issuing further calls before any observation comes back. This is a limitation of tool use, not of reasoning alone, because of where the remedy lies. Cuadron et al. find that sampling several trajectories and keeping the one that overthinks least improves outcomes, but concede that this selects among existing behaviors rather than changing them [29]. The intervention that does reach the behavior is a property of the tool interface. Models with native function calling overthink markedly less, and perform substantially better, than variants driven by prompted tool descriptions, which suggests the imbalance is in part an artifact of how tool use is elicited rather than a fixed disposition of the model.

Security and Robustness to Adversarial Tool Content A further limitation concerns the trustworthiness of the content a tool returns, rather than the tool call itself. Greshake et al. [44] show that LLM-integrated applications can be compromised via indirect prompt injection. Adversarial instructions embedded in retrieved or ingested content, such as a webpage or document, are treated by the model as legitimate commands rather than as data to reason over. INJECAGENT [168] extends this threat model to tool-using agents specifically, benchmarking whether an agent follows injected instructions instead of the user’s original task across attacks aimed at direct harm and at exfiltrating private data. Prompted REACT-style agents comply (i.e. follow the attacker’s injected instruction instead of the user’s original task) a substantial share of the time, and reinforcing that instruction with a dedicated hacking prompt roughly doubles the rate for GPT-4. Agents fine-tuned for function calling prove markedly more resistant, though the authors test only two such models and caution against generalizing from them. Resistance here is a byproduct of tool-use fine-tuning rather than of any objective aimed at injection. A related but distinct risk concerns the retrieved content itself rather than instructions embedded in it. Zou et al. [180] show that retrieval-augmented systems can be attacked by knowledge corruption, where injecting a handful of crafted texts per target question into a corpus of millions is enough to induce an attacker-chosen answer in the large majority of cases, with paraphrasing, perplexity filtering, and larger retrieval sets all failing to prevent it. The attack is demonstrated on FEVER and on a REACT-style agent as well, which places it squarely in the setting of automated fact-checking. Both surfaces, hijacking the model through injected instructions and corrupting the evidence it relies on, apply to any system whose tool-retrieved content forms the basis of its output.

These limitations, like those of reasoning in isolation, largely concern tool use as a standalone capability, independent of how it is embedded within a longer-running system.

2.3. Agentic Systems

In practice, neither reasoning nor tool use is typically deployed in isolation. Modern systems interleave the two repeatedly and often coordinate this process across multiple specialized agents to complete tasks that neither capability could accomplish alone. Such agentic systems make reasoning and tool use converge into a sustained, multi-step loop of planning, acting, and adapting to feedback.

Wang et al. [138] organize LLM-based agents into four modules: profiling, memory, planning, and action. This structure describes what an agent is assembled from without saying what separates it from the tool-using systems of the previous section, which also plan, act, and carry state within a single context window. Building on these components, we characterize an LLM agent by the closure of a perceive-reason-act loop over multiple iterations, where perception is the feedback the environment returns after each action. Two conditions make that loop agentic:

- *Persistent memory.* Memory must persist beyond the context of a single exchange, in the hybrid short- and long-term sense the same survey distinguishes from purely in-context state.
- *Run-time trajectory.* The length and content of the trajectory must be determined by the system at run time rather than fixed by the designer in advance.

The same survey also separates capability acquisition into fine-tuning and non-fine-tuning routes, naming the distinctive form of the latter *mechanism engineering*, which covers trial and error, experience accumulation, and self-driven evolution. Nearly every system surveyed below is an instance of it.

2.3.1. Single-Agent Systems

The boundary between tool use and agentic behavior is not strict. LATS [176] and CRITIC [41] already exhibit agentic characteristics, namely search over trajectories and critique grounded in external tool output, but in both the behavior resolves within a single episode. Persistence beyond the episode is what separates the systems discussed here, and its cheapest form costs almost nothing architecturally. REFLEXION [120] converts feedback on a completed trajectory, whether scalar or free-form, into a natural-language self-reflection, stores it in a bounded buffer, and conditions the next attempt at the same task on what it wrote. Mistakes are therefore corrected across trials rather than within one trajectory. The memory is deliberately designed, pairing a short-term trajectory with a handful of distilled long-term entries, yet everything in it serves the one task being retried and expires with it. That leaves open whether memory can be an object of design in its own right rather than a device for retrying.

MEMGPT [96] answers that question by detaching memory from any particular feedback loop and treating the context window itself as the resource to manage. Drawing an explicit analogy to operating-system virtual memory, it arranges storage into tiers: system instructions, writable working context, a FIFO queue, recall storage, and archival storage. The LLM issues function calls to search, write, and summarize across these tiers, while a queue manager watches for memory pressure and triggers eviction. The illusion is a context window far larger than the model’s actual one, achieved without touching the model. The cost is that reliability now rests on the model’s function-calling ability rather than on the architecture. The authors’ own numbers show how much that matters, since the same design reaches much higher performance on deep memory retrieval with GPT-4 compared to GPT-3.5.

What accumulates in MEMGPT is information, and information does not by itself make an agent capable of more. VOYAGER [137] stores competence instead, and in doing so also drops the fixed task that REFLEXION assumes. Its setting is Minecraft, an open-ended environment that supplies no terminal goal, so the agent must decide for itself what to attempt next. An automatic curriculum handles this, proposing each task from the agent’s current inventory, biome, and history of successes and failures. Code written to complete that task is refined over several rounds using environment feedback, execution errors, and a separate self-verification critic. Successful programs are then stored in a growing, embedding-indexed skill library rather than a log of lessons. Because the stored artifact is executable, competence transfers to tasks never attempted before, and ablating the library makes performance plateau in later stages.

Each of these systems widens what survives from one attempt to the next, and none of them asks whether a single attempt is well organized internally. DEPS [143] takes up that question in the same open-ended Minecraft setting as VOYAGER. An LLM planner produces an initial sequence of sub-goals for a long-horizon task. When one fails, a descriptor summarizes the resulting state in text, and the LLM uses that description to self-explain the failure before revising the plan, repeating this describe-explain-plan loop until the task succeeds. The loop alone does not suffice, because a corrected plan can still sequence independent sub-goals in an infeasible or wasteful order. A trainable selector therefore ranks parallel candidate sub-goals by their estimated steps to completion, grounding the ordering in a learned proximity signal rather than in the LLM’s judgment.

Read together, the four trace a widening of persistence, from repeated trials of one task, through arbitrary information within a session, to an open-ended sequence of self-proposed tasks. DEPS illustrates a limitation of this direction, since ordering the sub-goals of one plan is a problem no amount of accumulated history solves. It is also the only one of the four to reach for a trained module rather than a better prompt. None of them, however, requires more than a single agent acting in the environment.

2.3.2. Multi-Agent Systems

Multi-agent systems add a second axis along which persistence and specialization can occur. Instead of one model accumulating state across trials or tasks, several LLM instances interact within a shared environment, each instantiated as a distinct agent [78]. The motivation is that some tasks exceed what a single agent solves reliably, and decomposition across specialized roles is expected to close the gap. Whether it does is a question this section returns to at the end. The cost side is clearer, since more agents mean more inference calls and more communication, the same trade against reasoning quality that inference-time scaling makes earlier in this chapter.

How agents relate to one another varies along a small number of recurring structures. They may operate at the same hierarchical level with similar or opposing goals (equi-level), be coordinated by a central agent (hierarchical), combine both patterns (nested), or change roles and number over the course of a task (dynamic) [50]. Multi-agent debate [33] instantiates the equi-level case in its purest form. Several copies of an LLM independently answer the same question, then revise over multiple rounds after seeing one another's responses, converging through mutual critique rather than through any designated coordinator. The exchange requires no change to the underlying model or prompting method, and it reduces hallucination and reasoning errors by exposing each agent's claims to cross-checking it cannot perform on itself. The mechanism has a matching weakness, however, since debate optimizes for agreement rather than accuracy, and several copies of the same model can converge confidently on a shared wrong answer.

Planning and memory, both introduced for single agents above, acquire additional structure once several agents are involved. Global planning must balance three things: exploiting each agent's specific capabilities, keeping the workflow aligned with the overall goal, and managing the context each agent has to track [50]. Loops such as debate refine intermediate results before they are accepted. Within any single agent the same problem reappears from the inside, as the task of navigating layered context supplied by others rather than generated itself. Han et al. enumerate five memory types, of which short-term, long-term, external storage, and episodic memory all have single-agent analogues. Only consensus memory, knowledge shared and agreed upon across collaborating agents, has no counterpart in the systems of the previous section.

Every system described so far assumes that a designer fixed the topology and each agent's prompt in advance. That assumption is the recurring complaint in this literature, since hand-engineering both requires extensive trial and error and generalizes poorly to new tasks [84, 63]. A cluster of approaches responds by making the configuration itself something the system learns, using natural-language critique in place of a numerical gradient. TEXTGRAD [167] formalizes the move in its most general form. Given a pipeline of black-box components, an LLM runs a forward pass as usual, and a backward pass then has an LLM critique each intermediate output in natural language. An optimizer step updates the corresponding variable, mirroring backpropagation while operating over text and admitting non-differentiable objectives such as unit tests or simulator outputs.

Three lines of work narrow that general machinery onto multi-agent systems specifically. ANN [84] and EvoMAC [63] target topology and workflow, decomposing tasks across agents or layers of agents and attributing success or failure to individual agents through a gradient-like step. EvoMAC draws its signal from compiler and test results, so its attribution is grounded in the environment rather than in a critique. Replacing that objective feedback with LLM critique costs 12 to 22% performance, which puts a number on the difference between verifying against the environment and asking a model to judge its own output. Shen et al. [118] narrow the target further, first locating which agents underperform and why from communication trajectories, then optimizing only those agents' prompts instead of the system as a whole. AGENTIC CONTEXT ENGINEERING [172] changes the object rather than the scope, treating a system's shared context as what gets optimized.

Repeatedly compressing and rewriting that context erodes detail, so a generator, reflector, and curator instead add, update, and de-duplicate itemized entries in a persistent store, growing coverage rather than collapsing into brevity. All three approaches share a structural constraint. Optimizing a multi-agent system requires reading its full communication trajectory, so the optimizer’s own context window bounds the size of the system it can improve. Shen et al. use a fixed three-agent team (solver plus two reviewers) with one review round, and EvoMAC’s organizer prompt caps team size at roughly five by merging excess subtasks. Both systems are thus validated only at a scale small enough for hand-engineered coordination to remain feasible.

Each of these methods needs to know which agent caused what, and with several agents contributing jointly that is not given. The single-agent reflection mechanisms of the previous section never had to solve it. COPPER [13] does so by training one reflector shared across all agents and scoring it with a counterfactual reward, the difference between the system’s performance with and without a given agent’s reflection. Credit is thereby isolated per agent instead of being distributed uniformly from a single system-level reward. SAGE [79] takes a structural route, distributing work across a user, an assistant that acts, and a checker that provides feedback, iterating between assistant and checker until the output validates or an iteration limit is reached. A separate memory mechanism modeled on the Ebbinghaus forgetting curve decides which reflections persist in long-term memory and which decay, turning the bounded reflection buffer of REFLEXION into a continuously operating and selectively forgetting store.

Whether any of this makes multi-agent systems reliable is answered directly by Cemri et al. [19], who analyzed over 200 execution traces from seven frameworks. Failures divide into specification issues, misalignment between agents, and inadequate verification of task completion. The largest share (41.8%) originates in how the system was specified rather than in the underlying LLM’s reasoning capacity. Their intervention study is more pointed still. Prompt refinements produced gains ranging from under 1% to 9.4% across the two systems tested, and topology changes ranged from a non-significant 1% to 15.6%, the very levers the optimization methods above pull. Even the largest gains left most failure modes unresolved, leading the authors to argue that reliable performance will require more fundamental changes to agent organization, communication protocols, and verification, not just tactical prompt or topology fixes. Surveys of agentic systems in other application domains reach the same conclusion independently; work on automated claim verification, for instance, names dynamic task allocation and error detection robust enough to stop cascading errors as the central open problem for agentic pipelines specifically [136].

3. Related Work

Chapter 2 covered the capabilities an automated fact-checker is built from. This chapter covers the systems built out of them, and reads that literature with one question in view. When a system decides which checks to run on a claim, where does that decision come from?

Section 3.1 surveys automated fact-checking. Systems are grouped by the task scope they target, then compared on the modalities they support, the evidence sources they draw on, the verdicts and justifications they produce, and the benchmarks on which they report. Each grouping closes by identifying the origin of the verification procedure in the systems it covers, and Table 3.1 records that origin alongside the other three properties.

Section 3.2 turns to strategy learning, the body of work on acquiring reusable procedural knowledge for LLM-based agents without updating weights. Four mechanism families appear there (episodic memory, prompt optimization, workflow induction, and library learning), followed by the unsupervised topic modeling that the clustering stage of our own pipeline is built on. That survey is organized by the form a learned strategy takes, and each subsection then asks what data the strategy was learned from.

The two bodies of work answer the opening question in ways that never meet. A fact-checking system obtains its procedure from a prompt, from a designer, or from a set of weights, and none of those three routes accumulates anything across the claims the system sees. Strategy learning supplies exactly that accumulation, while assuming an agent that generates its own material and a task whose outcomes can be scored without a human. Section 3.3 states the gap that follows and the position this thesis takes on it.

3.1. Automated Fact-Checking

Multimodal automated fact-checking (MAFC) aims to replace or assist with the manual verification of factual claims using computational methods. Misinformation on social media and online news platforms spreads faster than professional fact-checkers can respond to it [34], which is why automated verification is an active area of research at the intersection of natural language processing, computer vision, information retrieval, and machine learning.

Akhtar et al. conceptualize the task as a three-stage pipeline: (1) claim detection and extraction, (2) evidence retrieval, and (3) verdict prediction with justification production [4]. This thesis assumes atomic claims as input, addressing the second and third stages. Existing systems within this scope differ along three dimensions, namely the task scope they target, the modalities they support, and the sources they draw evidence from. The following subsections address these differences, introducing each system once under task scope and referring back to it thereafter.

A fourth property cuts across all three and is not widely discussed in the literature. Regardless of its scope, modality support, or evidence source, a system must decide which checks to perform on a given claim. A

quote attribution and a manipulated photograph for example require different investigative approaches, and selecting the wrong approach allocates the retrieval budget to evidence that cannot resolve the issue. This section refers to this process as the *verification procedure*, and each subsection below concludes by identifying where the procedure comes from in the work it surveys.

Studies of professional fact-checking have produced consistent results. For example, Graves observed U.S. newsroom practices for four years and analyzed a single PolitiFact check. He described the verification process as having five stages: (1) choosing which claim to check, (2) tracing it to its origin, (3) consulting domain experts, (4) writing up results, and (5) disseminating the report [43]. Micallef et al. interviewed 21 fact-checkers from 19 countries and found that they all used the same five stages, though the tools and channels involved varied by region [90]. Graves adds that no single decisive test settles a contested political claim, so a verdict rests on triangulating evidence that is individually inconclusive.

Two features of this account matter for what follows. The routine is stable enough across organizations and countries to be treated as a describable object rather than as individual habit. Its decisive element is the checker’s choice of which sources to consult and in what order, which is exactly the choice an automated system also has to make. The remainder of this section shows that automated systems make that choice from their own priors, rather than from any record of how human checkers have made it before.

3.1.1. Task Scope

Systems diverge in the verification setting they target. General-purpose approaches aim to verify arbitrary claims end-to-end. Others narrow the problem to a specific misinformation type or a single domain, and a further line reformulates verification as retrieval against previously checked claims.

General-purpose fact-checking

General-purpose systems verify arbitrary claims end-to-end. They differ mainly in how they arrive at the verification procedure, and in the dominant paradigm the model plans its own checking process while it works.

DEFAME [16] was among the first systems to handle multimodal claims and multimodal evidence jointly, through a tool-augmented pipeline in which a single MLLM plans and executes retrieval and analysis actions drawn from a toolkit of web search, image and reverse image search, and geolocation. Having reasoned over the accumulated evidence, the model either commits to a verdict or returns to planning for another round, ending with a citation-backed report. The procedural knowledge sits in the planning prompt, which is written once and applies to every claim the system sees. LEMMA [157] follows the same principle in a single pass with a narrower action space, deciding first through a zero-shot assessment whether external evidence is needed at all. AGENTFACT [155] spreads the planning across separate planning, text-retrieval, image-analysis, reasoning, and explanation agents. Its planner emits a full plan rather than a next action, listing ordered reasoning steps, statements to validate, and search intents for the retrieval agents to work from. Retrieval and reasoning then alternate, with low confidence or an unresolved claim triggering another round and the previous trace returning to the planner. Sources are labeled reliable, unreliable, satire, or unsure before reasoning begins, and each evidence item keeps a stable identifier that the reasoning points cite, which makes the output auditable at the level of individual assertions.

Other systems replace model-directed planning with an explicit structure. T²Agent [30] runs training-free Monte Carlo Tree Search over a task-relevant subset of an extensible toolkit, decomposing verification into subtasks that target different forgery sources. PASS-FC [179] grounds each claim in a resolved time span and disambiguated entities, then issues structured queries against a filtered set of credible domains, restarting the loop through a reflection step when a verdict comes back contradicted or inconclusive. Actions are still chosen at run time in both, but the rule governing the choice is authored once and applies to every claim.

A further group fixes the structure completely, leaving no retrieval actions to select. DEBATECV [54] has two agents argue opposing stances over evidence retrieved in advance while a moderator weighs the exchange, and CRAVE [175] has an LLM write supporting and refuting rationales along four fixed aspects for a separate model to adjudicate. The structure only pays off once that judge is trained. LOCAL [83] instead decomposes the claim into verification and question-answering subtasks, then tests the assembled solution for logical equivalence to the original and for consistency under a flipped label.

A recurring problem across these systems is that self-directed retrieval readily admits noisy or unhelpful evidence. Noisy evidence degrades MLLM judgment actively rather than merely failing to inform it [148, 72]. Later systems therefore treat the evidence stream more deliberately. FACTCOMPASS [170] adapts CLIP for news-domain image retrieval, looks up the entities it detects in DBpedia and Wikidata for background, and filters redundant snippets, then reasons over the result with a trained model rather than a prompted one. Starting from the observation that images supplied indiscriminately can lower accuracy, AMuFC [66] splits the job between two vision-language models, one judging in natural language whether the claim needs an image, the other predicting veracity conditioned on that judgment alongside the retrieved evidence. LOKI [74] works on the input instead, decomposing text into atomic, checkworthy claims and captioning images before issuing per-claim queries. A hybrid design falls back on model-generated reasoning when retrieved evidence proves insufficient [146], an adaptation motivated by failures on emerging news.

The verification procedure in all of these systems is produced internally, whether by the model’s own planning, by a designer-authored search structure, or by training. It is improvised or fixed in shape, and in neither case does it carry claim-specific knowledge about which checks tend to settle which kinds of claims. Experience gained on earlier claims leaves no trace that later claims can draw on, so recurring failure modes persist without any mechanism for correction.

Specialized verification tasks

A second group narrows the task to a specific misinformation type, most prominently out-of-context (OOC) detection, in which a pristine image is paired with a misleading caption [1]. Fixing the type settles what has to be checked, because a repurposed image raises questions about origin and context and little else. Relieved of that question, this literature spends its effort on the components downstream of it, and divides according to which one it treats as the bottleneck. For the earliest of these systems the bottleneck was the model. SNIFFER [103] instruction-tunes an MLLM twice over, first onto news-domain entities and then onto OOC judgments, lifting an off-the-shelf backbone from near-chance to the best reported accuracy of its time. That gap set the agenda for what followed, and RAMA [160] answered it without tuning anything, fusing votes from three multimodal judges that read live-web evidence under different prompts.

Next, attention shifted to the evidence, on the argument that retrieval quality is the real constraint. E2LVLM [148] reranks retrieved snippets and rewrites the best one into coherent context before the model sees it. CMIE [72] questions what the evidence is matched against, having the model first articulate why image and text might legitimately belong together, then scoring snippets against that relationship rather than against the

raw pair. Web noise can also be avoided rather than filtered, and EXCLAIM [151] does so by retrieving from a curated index over entities and events, though it builds that index from the training split of the benchmark it reports on. The strongest form of the argument is COVE [132], which reconstructs the image’s context across seven structured items and turns to the caption only afterwards. A third group takes the pipeline as given and attacks what it costs to build. HYBRIDNET [127] extracts reasoning signals from a frozen MLLM and trains a small classifier over them, matching supervised baselines on a fraction of the labels while leaving inference expensive. MULTI-MLLM [46] distills two large teachers into a small student and annotates only the cases where they disagree, cutting both annotation and inference cost. FACT-R1 [169] stands apart from all of them, abandoning retrieval and moving to news video, where long chain-of-thought tuning, preference alignment, and GRPO teach the model to name the manipulated entity on its own. All of these gains rest on a narrow base, however, since nearly every system above is evaluated on the same synthetic split of NEWSCLIPPINGS [82]. The one exception is COVE, which is also tested on real-world OOC cases, where its F1 score falls by more than thirty points.

These systems all know in advance that the image itself is authentic, so none of them needs to test for manipulation. The order of the checks that remain is another matter. COVE reconstructs the image’s context first and turns to the caption afterwards, while SNIFFER and E2LVLM go straight to the caption and test it against retrieved evidence. Nothing in the task definition picks between these two orders. COVE’s ablations show that context first works better for OOC claims, but that came out of running the experiment, not out of the definition. The verification procedure here therefore comes from its designers. A general-purpose system would need one such finding for every kind of claim it might receive. Narrowing the task is what made COVE’s finding possible, and that is the move a general-purpose system gives up.

Domain-specific fact-checking

A third group narrows the domain rather than the misinformation type. Climate and health each come with a recognized body of authoritative literature, so which sources to trust is settled before verification begins. Knowing the domain also changes what a system can produce, and several of these write a correction rather than return a label. CLIMINATOR [71] runs several advocate agents, all but one of them retrieving from a corpus of its own such as IPCC or WMO reports, and has a mediator reconcile their verdicts. The debate is largely dormant, however. Across 170 claims the advocates disagreed on one, and only an added advocate grounded in denial literature raised that to eighteen. The CLIMATECHECK shared task [2] strips the problem down instead, splitting it into two steps: retrieving abstracts from a corpus of nearly 400,000 climate papers, then verifying the claim against what was retrieved. Shahbazi and Behnamian [116] reverse that order. A classifier trained on a corpus of already-debunked climate claims flags the incoming claim, and a similarity check catches paraphrases of known falsehoods. The verdict therefore rests on resemblance to past misinformation rather than on retrieved evidence. Only flagged claims proceed, and keyword and embedding-based search over IPCC reports and peer-reviewed literature supplies the material for the rebuttal. Evidence shapes the correction without ever revising the label.

Health systems commit to their corpus just as firmly. TRUMORGPT [51] builds a knowledge graph of subject-relation-object facts drawn from medical news and DBPEDIA, then treats a claim as true when a matching path through the graph exists. Claims that fall outside the graph return no verdict at all. The counterspeech pipeline of Anik et al. [5] is the exception here, pairing a curated medical knowledge base with live web search before summarizing, drafting, and refining a reply.

Restricting the domain settles what counts as adequate evidence, but it does not settle how to check. The three climate systems above share a domain and draw on the same kind of scientific corpus, yet one stages

an advocate debate, one retrieves abstracts and then classifies, and one classifies before retrieving and then writes a rebuttal. Nothing about climate claims picks among those three shapes. As with the specialized tasks, the shape came from the designers, and there is no reason it should carry to a domain they did not have in mind.

Reuse of prior fact-checks

Claim matching, also called detection of previously fact-checked claims, formulates verification as retrieval [115]. Given an incoming claim, a system ranks a pool of already verified claims by how useful each would be for checking it, and a human decides what to do with the ranking. The task has been a recurring track at the CheckThat! lab [93], where it is scored with standard ranking measures rather than by verdict accuracy. CLAIMCHECK [101] puts it to work inside a full pipeline, searching roughly sixteen trusted fact-checking sites for an existing article before falling back on question generation and web retrieval for whatever it cannot match. In most of the literature fact-check articles supply labels for training or evaluation, whereas here they supply the answer.

A matched claim needs no evidence gathering and no reasoning over what is gathered, which is the appeal. The limitation follows from the same mechanism. Matching returns the article written about one specific claim, and misinformation recirculates in variants that defeat surface similarity. CLAIMCHECK finds an existing fact-check for roughly one claim in six, which is the practical measure of how narrow the mechanism is. A claim about a different flood, a different politician, or a different date falls outside the pool even when the routine that would settle it is identical to one a human already carried out.

A fact-check article therefore carries two transferable things of unequal reach. The verdict transfers to the claim the article was written about and to near-repeats of it, which is genuine but covers little. The procedure reaches further. An article records which sources the checker consulted, in what order, and which step proved decisive, and that record applies to any claim of the same shape. The content of fact-check articles has also been used to build benchmarks, some of which keep the reasoning alongside the verdict (see 3.1.5). What those benchmarks put to work is still the verdict and the justification, both as targets for scoring systems rather than as guidance for checking a new claim. The record is collected, kept, and then set aside.

3.1.2. Modalities

Supporting different input and evidence modalities changes the design of a fact-checking system, since claims and evidence that are textual, visual, or mixed call for different retrieval, representation, and reasoning strategies. Table 3.1 gives the modality support of every system introduced above.

Early systems were built for textual claims and textual evidence [130, 48], formulating verification as natural language inference or evidence reasoning. Initial claim classification methods applied supervised text classification, learning veracity from surface linguistic patterns without genuine evidence grounding [130]. Recent text-only systems differ in where they concentrate effort. Some work on the reasoning, whether through competing rationales [175], consistency checks over a decomposed claim [83], or multi-agent debate [54]. Others work on retrieval, grounding each claim temporally and to specific entities before searching [179], or on deployment cost, breaking verification into subtasks a small model can handle [101, 146].

The prevalence of image-based misinformation motivated multimodal approaches [34], and two architectural paradigms emerged. One converts visual input into textual descriptions and reduces multimodal verification

to text-based reasoning. LOKI [74] takes this route, captioning images and sampling every fiftieth video frame through a vision-language model before verification begins, which lets an existing text pipeline be reused at the cost of discarding fine-grained visual detail. The other employs vision-language models operating on joint representations, so that evidence in one modality can shape the interpretation of the other, as in DEFAME [16] and FACTCOMPASS [170].

Asymmetry between the claim side and the evidence side is common. AMuFC [66] verifies textual claims against multimodal evidence, whereas RAMA [160], CMIE [72], MULTI-MLLM [46], and LOKI [74] accept visual claims but retrieve textual evidence only. Systems that are multimodal on both sides remain a minority, as they were when Abdelnabi et al. [1] first paired open-web image and text evidence for out-of-context claims, though DEFAME [16] and FACTCOMPASS [170] have since carried the setting beyond that one task.

Modality support is not always a property of the method. In pipeline systems the backbone sets what can be read at all, and preprocessing supplies the rest. DEFAME [16] samples frames when its backbone cannot read video and passes the whole file when a newer one can, so the same method occupies two different rows of Table 3.1 depending on what it is run with. LOKI [74] does the same at a fixed rate of one frame in fifty. The table therefore records what a system accepts rather than how deeply it processes what it accepts.

Video and audio remain rare all the same. DEFAME [16] and LOKI [74] accept video claims, and DEFAME alone retrieves video evidence. FACT-R1 [169] differs in kind rather than degree, trained on video-text data rather than adapted to video, and drawing an additional signal from the audio track, though it retrieves no external evidence at all. The gap matters because short-form video is now a primary vector for misinformation [102], and a checker restricted to text and images cannot verify most of what circulates on the platforms where claims originate.

3.1.3. Evidence Sources

The choice of evidence source shapes which claims can be verified, how current and reliable the verdicts are, and how much noise the reasoning module must absorb.

Some systems forgo retrieval entirely [92, 169], which removes retrieval latency and infrastructure complexity along with the auditable evidence trail. Each inherits a different limit in exchange. FACT-R1 leans on knowledge held in the weights of a model that is poor at recognizing its own cutoff [76], while CLAAF learns discriminative features from a single curated dataset and cannot reach past its scope. The most controlled retrieval setting is a curated corpus. Domain-specific systems favor it because their claims are narrow enough for a bounded evidence base, and grounding verification in scientific publications [71, 2, 116] or structured knowledge graphs [51] prioritizes provenance over breadth, though some pair a curated base with live search rather than choosing between them [5]. Wikipedia occupies an analogous position for general-domain text-only verification [175, 83]. Coverage and freshness limit both settings, since corpora require maintenance, tend to be English-centric, and fail outside their scope, while Wikipedia leaves gaps for recent events and non-Western topics [132].

Open-web retrieval removes the coverage ceiling and introduces a noise problem in its place. Systems querying the live web can in principle verify any claim against current sources [16, 157, 30, 155, 160], and agentic pipelines exploit this by refining queries across rounds [16, 30, 155] or expanding them across languages [179]. The web also carries unreliable [25], contradictory [48], and adversarially crafted [80] content alongside authoritative sources, so verdict quality depends heavily on filtering and ranking. A subtler hazard is leakage. Retrieved pages frequently turn out to be fact-check articles about the claim under test, which inflates benchmark scores without improving verification of anything new [25]. Both problems have motivated

System	Claim Modalities	Evidence Modalities	Evidence Source(s)	Procedure
<i>General-purpose</i>				
DEFAME [16]	Text, Image, Video	Text, Image, Video	Open web	Model-directed
LEMMA [157]	Text, Image	Text, Image	Open web	Model-directed
LOKI [74]	Text, Image, Video	Text	Open web	Model-directed
AMuFC [66]	Text	Text, Image	Corpus + web	Model-directed
AGENTFACT [155]	Text, Image	Text, Image	Open web	Model-directed
Wei et al. [146]	Text	Text	Hybrid	Model-directed
T2AGENT [30]	Text, Image	Text, Image	Open web	Search structure
PASS-FC [179]	Text	Text	Open web	Search structure
DEBATECV [54]	Text	Text	Corpus	Debate
CRAVE [175]	Text	Text	Wikipedia	Rationale compare
LoCAL [83]	Text	Text	Wikipedia	Rationale compare
CLAAF [92]	Text, Image	—	Parametric	Architecture
FACTCOMPASS [170]	Text, Image	Text, Image	Open web	Trained (RL)
<i>OOO detection</i>				
Abdelnabi et al. [1]	Text, Image	Text, Image	Open web	Task-fixed
COVE [132]	Text, Image	Text, Image	RIS + Wiki.	Task-fixed
E2LVM [148]	Text, Image	Text, Image	Corpus	Task-fixed
CMIE [72]	Text, Image	Text	Open web	Task-fixed
RAMA [160]	Text, Image	Text	Open web	Voting
EXCLAIM [151]	Text, Image	Text, Image	Corpus	Voting
SNIFFER [103]	Text, Image	Text, Image	Open web	Trained (SFT)
FACT-R1 [169]	Text, Video, Audio	—	Parametric	Trained (RL)
HYBRIDNET [127]	Text, Image	Text, Image	Open web	Trained (distill.)
MULTI-MLLM [46]	Text, Image	Text	Corpus	Trained (distill.)
<i>Domain-specific</i>				
CLIMINATOR [71]	Text	Text	Corpus	Debate
CLIMATECHECK [2]	Text	Text	Corpus	Search structure
Shahbazi and Behnamian [116]	Text	Text	Corpus	Prior fact-checks
TRUMORGPT [51]	Text	Text	Corpus (KG)	Search structure
Anik et al. [5]	Text	Text	Open web	Search structure
<i>Claim matching</i>				
CLAIMCHECK [101]	Text	Text	Open web	Prior fact-checks

Table 3.1.: Fact-checking systems by task scope, modality support, evidence source, and the origin of the verification procedure. RIS = reverse image search.

dedicated evidence-handling stages before the reasoning module sees anything, whether reranking and rewriting snippets [148], scoring relevance against a generated image-text relationship [72], classifying source reliability [155], or pruning retrieved text for redundancy [170].

Neither modality support nor evidence source determines the verification procedure. LEMMA [157] and T2AGENT [30] accept the same claim and evidence modalities and draw on the same open web, yet LEMMA lets the model plan as it goes and T2AGENT runs a tree search over its tools. The last column of Table 3.1 records that difference.

3.1.4. Verdicts and Justifications

The third pipeline stage has received less attention than retrieval. Early work framed justification production as a summarization problem over the retrieved evidence [10], and current systems have largely absorbed that framing into generation, emitting a categorical label alongside a natural-language rationale. DEFAME produces a citation-backed report [16], and AGENTFACT dedicates a separate agent to explanation [155]. Attention has since moved to whether those rationales can be trusted. Human protocols now score them for unsupported inference and for misused or fabricated evidence [155], and one study measures usefulness instead, finding that a system’s explanations help a second model correct cases it had got wrong [170]. COVE pushes furthest in this direction, showing that its reconstructed context remains useful to human annotators judging new captions about the same image [132], which makes the output an artifact that outlives the verdict it was produced for.

Categorical labels sit awkwardly with real claims. A claim can be accurate in substance while the image beside it is recycled from another event, and a coarse label collapses the two into one judgment. The field has settled on no particular granularity. FACTCOMPASS reduces veracity to binary real or fake because of its training data [170], CLIMINATOR evaluates at four levels ranging from twelve categories down to two [71], and RW-Post keeps a separate challenge set for the mixture and outdated cases its three-way scheme cannot express [155]. Finer labels are harder to predict as well, since TRUMORGPT loses nearly forty points of accuracy when moved from a binary split to PolitiFact’s six categories [51]. VERITAS [110] answers with five disentangled continuous scores in place of a bin, covering media authenticity, media contextualization, veracity, context coverage, and integrity, scored by mean squared error (see 3.1.5). This changes what a system must produce, since it must now express calibrated uncertainty along several axes instead of choosing a category.

3.1.5. Benchmarks

Datasets have shaped the field’s trajectory. FEVER [131] established claim verification against Wikipedia, requiring a system to predict SUPPORTED, REFUTED, or NOT ENOUGH INFO and to supply sentence-level evidence, and its baselines already showed retrieval rather than classification to be the bottleneck. LIAR [140] grounded the task in political discourse with 12.8K PolitiFact statements and six-way labels, but supplied no machine-readable evidence, which confined systems to the statement text and to speaker metadata that added almost nothing over it. Neither admits non-textual claims or non-textual evidence.

Open-domain and multimodal benchmarks addressed these limits in turn. AVERITEC [113] moved verification onto the open web through question-answer decomposition with cited answers. MOCHEG [161] extended evidence retrieval to images while keeping claims textual, and AVERIMATEC [18] closed that gap by applying QA decomposition to real image-text claims under temporally constrained evidence, again finding retrieval to be the dominant failure mode. RW-Post [155], the benchmark accompanying AGENTFACT, instead makes evidence use auditable. Extracted from 13,000 Snopes articles, each instance aligns a claim with the social-media post that carried it and attaches a reasoning trace whose individual points cite specific evidence items. Scale followed along two axes. MMM-FACT [156] contributes over 125K claims across three decades with retrieval-difficulty tiers, while M4FC [38] emphasizes multilinguality and task diversity across ten languages. A further set of benchmarks narrows scope instead, targeting specific manipulation types [82, 98, 139, 81, 158, 77] or specific domains and languages [134, 109, 57, 102]. None of these narrower benchmarks targets the general, open-domain, multimodal setting this thesis addresses.

A more pressing problem cuts across all of them. Static benchmarks leak into the pretraining data of the models they evaluate, inflating reported performance. Two recent benchmarks answer by refreshing their contents

rather than fixing them once. XFACTA [152] collects contemporary social-media cases and keeps them current through a semi-automatic loop in which detectors help surface new material. VERITAS [110] runs a fully automated pipeline on a quarterly cycle, ingesting claims from over a hundred fact-checking organizations across 54 languages and covering text, images, and video. Evaluating current MLLMs on recent VERITAS splits shows error rising sharply immediately after each model’s knowledge cutoff, which confirms that older benchmarks overstate real verification ability. VERITAS serves as the primary evaluation benchmark for this thesis, for its update cadence, its multilingual and audiovisual coverage, and the disentangled continuous scores described above.

Because VERITAS is built from published fact-checks, every claim in it arrives paired with the article that resolved it. The record described in 3.1.1 is therefore already available for the claims this thesis evaluates on, and no harvesting pipeline forms part of what follows.

3.2. Strategy Learning

Learning reusable strategies for LLM-based agents without updating model weights has become an active research direction, but proposed mechanisms differ substantially in *what form* the learned strategy takes and, more importantly for our purposes, *from what data* it is learned. We organize prior work along the first axis before returning to the second, which motivates our contribution directly.

By the form the learned strategy takes, the agent-learning work falls into four strands:

1. *Reflection and episodic memory* keeps strategies as natural-language memory that accumulates across tasks and is curated as it grows, extending the per-episode reflection of Section 2.3.1 beyond the task being retried.
2. *Prompt optimization* treats a single prompt as an explicit optimization target and refines it through scored candidate search.
3. *Workflow and system design* learns structured multi-step workflows or agent designs, typically via execution-guided search.
4. *Library induction* also targets structured, reusable artifacts, but discovers them by compressing a corpus of individual task solutions into libraries of composable functions.

We close with unsupervised topic-discovery methods, which sit outside the agent-learning literature but underpin the clustering stage of our own pipeline.

These strands differ in mechanism yet converge on where the learning signal comes from. Almost without exception, the methods we survey bootstrap from data the agent or model produces itself: self-play trajectories, LLM-generated candidate solutions, and programs synthesized by the model. Where an external corpus does enter, it supplies input–output pairs that say nothing about how the task was solved. Our setting inverts this assumption. Both of our learning mechanisms, the single-file strategy synthesizer and the clustering-based blueprint pipeline, consume a fixed corpus of human-authored fact-checking articles, treating expert demonstration rather than agent experience as the source from which strategy is derived.

3.2.1. Reflection and Episodic Memory

REFLEXION’s memory is built to retry one task, and it expires with it (Section 2.3.1). The three systems here keep the reflective loop but change what the memory is for, extending it across tasks and, eventually, into a standing artifact of its own. EXPEL [174] pools trajectories gathered via REFLEXION-style retries and distills them into cross-task insights through explicit add, edit, upvote and downvote operations, and retrieves similar past successes as in-context exemplars at inference. Its insights transfer to FEVER [131], making it the closest of this group to our own domain. AGENT-PRO [173] moves reflection from the single episode to the policy, distilling whole game trajectories into standing behavioral guidelines and world-modeling instructions. Because these candidates are verified by replay and selected by depth-first search against a payoff objective, AGENT-PRO sits as much in the optimization family of Section 3.2.2 as in this one. DYNAMIC CHEATSHEET [126] pushes the mechanism furthest, maintaining one persistent memory artifact that is curated rather than appended to after every query, with no ground-truth signal at any point. Its cumulative variant closely mirrors the incremental-fold structure of our single-file strategy synthesizer.

These four share a limitation for our setting. The material being distilled is exclusively self-generated, whether agent rollouts, evaluator-verified successes, or, in DYNAMIC CHEATSHEET’s case, unlabeled answers to the target query. None draws on pre-existing human expert process traces. Our single-file strategy pipeline operates in that regime, folding a body of human-authored fact-checking articles into the evolving strategy document.

3.2.2. Automatic Prompt and Instruction Optimization

The prompt itself becomes the object being learned once reflection gives way to an explicit scoring loop. PROTEGI [100] draws the analogy to gradient descent directly. An LLM criticizes a prompt’s failures on a minibatch, the resulting textual gradients are turned into candidate edits, and beam search with bandit-based best-arm identification selects among them. One of its four benchmarks is LIAR [140], so the method has already been pointed at prompt design for claim verification. OPRO [159] drops the hand-designed edit operator, feeding the optimizer LLM a meta-prompt of past instructions with their scores and letting it propose improvements directly. APE [178] moves one step earlier, synthesizing the first instruction from a handful of input–output demonstrations and selecting by execution accuracy instead of refining a seed.

INSTRUCTION INDUCTION [58] sits apart from these three. It runs no search, prompting a model once to state the rule that fits five demonstrations, and its contribution is the finding that this ability emerges only in large, instruction-tuned models. That primitive, moving from worked examples to an explicit description of the task, is the one both of our synthesis stages depend on.

The three optimizers require an explicit objective and a candidate pool at every step, where our single-file synthesizer performs one sequential edit per batch with neither. The difference is forced by the data rather than chosen for simplicity. A scalar score presumes outcomes that can be graded automatically, and the OPRO authors themselves note that aggregate accuracy is thin feedback. A corpus of expert verification articles offers no such signal, since nothing grades how closely a synthesized strategy matches the reasoning an expert actually applied.

3.2.3. Automated Agentic Workflow and System Design

What this strand learns is structure, in the form of multi-step workflows and code graphs that resemble the verification blueprints our clustering-based system produces. AGENT WORKFLOW MEMORY (AWM) [145] induces reusable web-navigation workflows from trajectories, and is notable here for its offline mode, which operates over pre-collected task demonstrations rather than self-generated rollouts. That is the closest precedent we found to learning from a fixed corpus rather than from interaction. [171] and ADAS [62] instead treat the workflow or the agent design itself as a search problem, the former running Monte Carlo Tree Search over code-represented workflows, the latter having a meta-agent program successive agents against a growing archive of designs. Neither progresses without repeated execution and scoring on a validation set. AFLOW also constrains its search with a fixed set of reusable operators, which is the part of that design closest to composing a blueprint from a bounded vocabulary of verification steps.

The ADAPTIVE-SOLVER framework [177] learns no new strategies at all. It arranges a pool of existing solvers into a cascade ordered by increasing cost on a validation set, then escalates from one solver to the next whenever answer consistency suggests the current solution is unreliable. Difficulty is never estimated for the problem at hand in advance, only revealed by failure. We return to this mechanism in Section 4.2, where our own selection stage gates a cheap probe on its confidence and falls back to an LLM for the remainder. Apart from AWM’s offline mode, structure in this strand is discovered through execution and search rather than read from a static corpus.

3.2.4. Tool and Library Induction

Where a workflow fixes an order of steps, a library fixes the pieces and leaves the order to whatever calls them. This strand builds such libraries by compressing solved tasks. DREAMCODER [35] set the pattern with a wake-sleep Bayesian loop. Its wake phase searches for programs that solve each task. A first sleep phase refactors those solutions via version spaces, extracting common subprograms into a growing symbolic library. A second sleep phase retrains a neural recognition model on replayed and imagined tasks, so search stays tractable as the library deepens. STITCH [14] keeps the compression objective but abandons the deductive search that produced it, synthesizing abstractions top-down under corpus-guided pruning. Syntactic matching and branch-and-bound bounds make it three to four orders of magnitude faster than DREAMCODER’s refactoring, at two orders less memory, with comparable or better compression. LILO [42] layers an LLM onto this symbolic core in two places, guiding the initial program search and, once STITCH has compressed, naming and documenting each learned abstraction. Removing that documentation measurably degrades the synthesizer’s ability to deploy the library it helped build.

A second cluster applies the same compress-and-reuse logic to LLM-generated programs, dropping the wake-sleep scaffolding for lighter, execution-verified loops. REGAL [122] groups related examples into batches, refactors each batch into shared helper functions, retries failed refactorings against execution feedback, and periodically prunes or edits helpers that fail to generalize. Its grouping step is the nearest analogue in this literature to the clustering stage of our own pipeline, where related articles are collected before any abstraction is written. TROVE [144] runs the idea as a stream. For each incoming task it samples solutions in three modes: importing an existing tool, creating a new one, or falling back to primitives. It then selects among them by execution agreement and simplicity. The shared toolbox is trimmed periodically, so only functions that earn their reuse survive. That grow-then-consolidate rhythm matches the merge-and-consolidate pass in our clustering-based pipeline. VOYAGER’s skill library [137] accumulates and reuses in an embodied setting, admitting only programs that pass a self-verification critic and indexing them by embedding similarity

for later retrieval and composition, with nothing compressing or trimming the library as it grows. CREATOR [104] takes the loosest form, synthesizing a bespoke tool per task and keeping no shared library, in order to separate the abstract reasoning of deciding what tool is needed from the concrete reasoning of applying it.

In nearly all of these systems the corpus being compressed is one the system produced. DREAMCODER compresses what its own wake phase discovers, and TROVE, LILO and VOYAGER reuse the model’s own solutions. STITCH is the informative case. As a compression algorithm it is indifferent to where its corpus came from, and it has been run on pre-existing program corpora, so the machinery for extracting abstractions from externally authored material already exists. It has not been pointed at human process traces. Our clustering-based pipeline applies the same discover-and-consolidate pattern to a corpus the system never produced, a body of human-written fact-checking articles.

3.2.5. Unsupervised Structure Discovery

The clustering stage of our pipeline is built on the topic-modeling recipe of BERTOPIC [45]: documents are represented with pretrained sentence embeddings, reduced in dimensionality with UMAP, grouped with HDBSCAN, and given a per-cluster representation through class-based TF-IDF. Grouping by embedding rather than by word co-occurrence is what separates it from classical probabilistic topic models, and it lets clusters track semantic similarity between documents that share little vocabulary. Two properties matter for any downstream use. HDBSCAN leaves low-density documents unassigned, and each document is assigned to a single topic.

We adopt the embed–reduce–cluster stages to group fact-checking articles by claim type, and replace class-based TF-IDF with an LLM-driven synthesis of a verification blueprint per cluster. A ranked list of terms can label a cluster but cannot state an ordered procedure. Chapter 4 takes up what the two properties above imply for blueprint coverage and selection.

3.3. Research Gap

Table 3.1 records where each system’s verification procedure comes from. Apart from claim matching, which reuses a stored verdict rather than deriving a procedure at all, the table’s entries collapse into three families that share a property. A model-directed planner writes the procedure from its own priors while it works. A designer-fixed structure encodes the priors of the people who built it. Trained systems do derive theirs from data, though the signal that shapes it is the outcome of a check rather than any account of how one was carried out. On none of the three routes does the procedure come from a record of how this kind of claim has been checked before.

Priors are a poor basis for this particular choice. A plan a model produces is unreliable in ways its own trace does not disclose [133, 70], and a procedure that holds on one phrasing of a problem often fails on a superficial variant of it [91]. The cost lands where the field can least afford it, since retrieval is the dominant failure mode on open-domain benchmarks [131, 18]. AVERIMATEC [18] gives the size of the shortfall, with baselines reaching evidence recall between 0.12 and 0.25 while the same verifiers score 0.78 to 0.82 on gold evidence. Which checks a system decides to run is what determines which queries it issues, so a misjudged procedure spends the retrieval budget on evidence that cannot settle the claim.

A record of how the choice has been made well does exist, and the field already holds it. Newsroom studies describe a stable routine whose decisive element is the choice of which sources to consult and in what order [43, 90], and a published fact-check article documents that choice for a claim someone already settled. Such articles are collected in quantity and read for their verdicts, as training labels, as evaluation targets, or as match candidates [101, 155, 110]. Claim matching shows how far a verdict reaches on its own. CLAIMCHECK [101] resolves matched claims at 75.9% accuracy against a 62.6% system average, yet finds a match for only 15.8% of claims, because a verdict is true of one claim while the routine behind it holds for a whole kind of claim.

Section 3.2 surveys the machinery that could read an article for its routine instead, and every instance of it was built for a different supply of data. Episodic memory distills agent rollouts [174, 173], prompt optimizers rank candidates against a metric [100, 159], and workflow and library induction compress solutions the system produced itself [171, 35, 122, 144]. The latter two also depend on automatic grading, which a corpus of articles cannot supply, since nothing runs a synthesized procedure against the article that inspired it and returns a score. Two cases approach the edge of the pattern. AGENT WORKFLOW MEMORY [145] induces workflows offline from pre-collected demonstrations, and STITCH [14] compresses program corpora it did not generate. Neither has been pointed at a record of how a human solved the task.

The gap concerns the input to learning rather than the amount of it. Verification procedures are derived from model priors, from designer intuition, or from outcome supervision, and the one available record of expert practice is read only for the labels it carries. No existing work extracts from that record the procedure it documents, at the level of claim types rather than individual claims, in a form a system can select from at inference time.

This thesis takes that record as its input and produces *verification blueprints*, ordered procedures written in natural language, each synthesized from a group of articles that verified claims of the same kind. Relative to prior work, two things change: the *material* a procedure is derived from, and the *form* the result takes. The material is a corpus of human fact-check articles, read for the procedure it documents rather than for its labels. The form is an *interpretable* artifact rather than a set of weights. Because a blueprint is an ordered procedure in natural language, it can be read, audited, and revised before deployment, instead of inspected afterwards through a trace that may rationalize the conclusion it accompanies [3]. Because each blueprint describes a kind of claim rather than a single one, it reaches where the verdict a matched article carries cannot. Synthesis is a one-time offline pass over a fixed corpus, so the system does not learn while it runs, and no claim it checks alters what it does on the next. Chapter 4 develops the two synthesis routes and the selection stage that decides which blueprint an incoming claim receives.

4. Methods

This chapter introduces a multi-agent system that verifies a claim by gathering evidence for it, and the procedure that produces the verification strategies the system runs on. A top-level agent leads each investigation. It delegates evidence retrieval to specialized worker agents, one for the open web and one for attached images and video, and a judge assigns the final label once enough evidence has been collected. Figure 4.1 gives the overview.

The top-level agent does not carry the strategy. The control logic that separates, for example, the verification of a quote attribution from the verification of a photograph is expressed as data rather than as branching code inside the agent. That data is a *blueprint*, and exactly one is selected per claim. This is the central design decision of the work, and two properties follow from it. A strategy becomes inspectable and auditable independently of the runtime that executes it. It also becomes a manipulable object, one that can be synthesized, merged, and evaluated automatically without modifying any agent.

The chapter follows those two properties. Sections 4.1 and 4.2 define the blueprint representation and how an incoming claim is routed to one. Section 4.3 describes the agents that execute it. Section 4.4 then treats blueprints a learning subject, deriving a pool of them from a corpus of professional fact-checks. Finally, Section 4.5 defines a baseline that holds every agent fixed and replaces the pool with a single free-text document, isolating what the structure contributes.

4.1. Blueprints

A blueprint is a declarative specification of a verification strategy for one class of claim. It is a schema-validated data structure that the orchestration agent consumes at runtime, not executable code and not a prompt template. For an incoming claim, exactly one blueprint is selected. The agent does not run the blueprint as a program. It reads the declared workflow as guidance and an LLM planner decides the concrete work at each step, delegating that work to specialized worker agents. The contrast with code is where control sits. Code fixes both what happens and that it happens, whereas a blueprint fixes only the order of investigation and leaves how each step is carried out to the planner (elaborated at the end of this section).

A blueprint comprises six components, summarized below.

Identity (name & description) The description is the blueprint’s self-advertisement to the selector, and is read by a language model when rule-based filtering leaves more than one candidate. The learning pipeline treats it accordingly, revising descriptions to sharpen the boundary between blueprints that compete for the same claims.

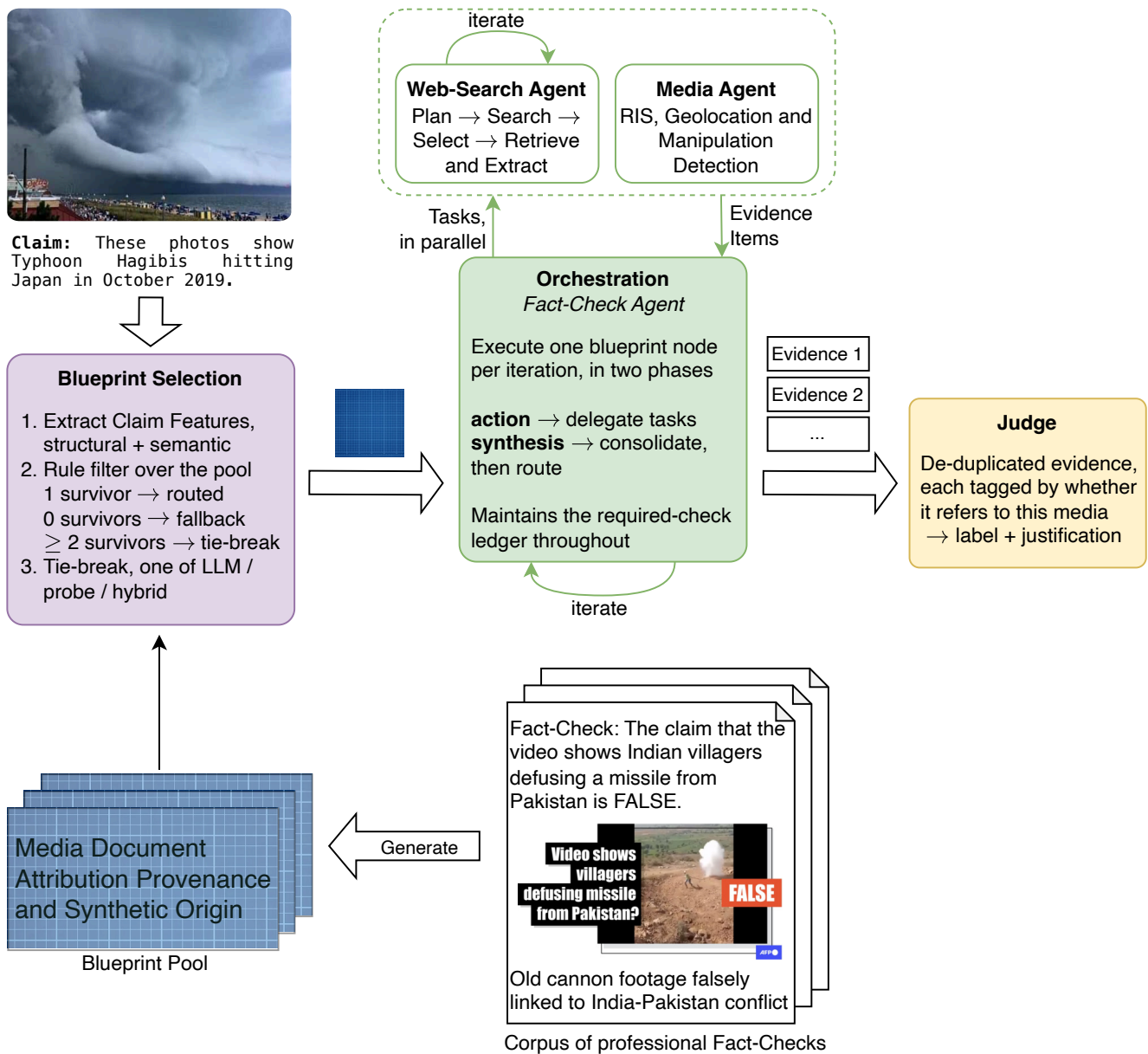


Figure 4.1.: High-level pipeline of the verification process. An incoming claim is routed to exactly one blueprint from the learned pool. The orchestration agent executes that blueprint one node per iteration, delegating retrieval to the web-search and media agents and tracking which required checks remain open. The loop ends when the planner finalizes, when routing reaches `finalize`, when a node has no outgoing transition, or when the iteration budget is spent. Assembled evidence then passes to the judge, which is the only component that knows the benchmark's label space. Blueprints are produced offline, from the corpus of professional fact-checks shown at the bottom, by the learning pipeline of Section 4.4.

Image taken from <https://factcheck.afp.com/doc.afp.com.46U349D>

Entry Conditions A conjunctive `all` list and a disjunctive `any` list of predicates of the form `{feature, op, value}` (e.g. `has_image==true`), evaluated against a feature vector extracted from the claim. Every predicate in `all` must hold and at least one predicate in `any` must hold if the list is non-empty. Supported operators cover equality, ordering comparisons, and membership. Entry conditions bind hard, i.e. a blueprint whose conditions fail is ineligible for the claim, and no other signal can override that. The rationale is that some eligibility facts are not judgment calls, e.g. a workflow built entirely around reverse image search must never be selected for a text-only claim.

Selector Hints Where entry conditions establish eligibility, hints express preference among the eligible. Positive examples are claims the blueprint should win; negative examples are claims it should concede to another blueprint. Several blueprints routinely satisfy the hard filter for the same claim and the hints supply the signal that separates them. Hints affect ordering only, and can neither admit a blueprint the hard filter rejected nor exclude one it admitted.

Policy Constraints The capabilities available to a run: (1) the worker agents that tasks may be delegated to, (2) whether an explicit search for counter-evidence is required, and (3) the maximum number of iterations. The iteration budget does more than cap cost, since it also determines how much of the graph a run can reach. It is therefore subject to a floor, derived in Section 4.4.6 and enforced whenever a blueprint is loaded.

Required Checks An identified set of conditions that should be settled before a verdict is issued. Each is tracked across the run through the states *unchecked*, *supported*, *refuted*, and *unclear*, and is re-evaluated as evidence accumulates. Checks constitute the blueprint’s quality contract as they express what the strategy considers a complete investigation, stated independently of the particular route by which the workflow reaches that state. Checks are either global or node-scoped and the scoping mechanism exists to support strategies with multiple alternative investigative paths.

Verification Graph A designated start node and a set of nodes of two types, connected by conditional edges.

Action nodes dispatch work to worker agents. Each action names the agent, an `intent` stating what the step should accomplish, and `query_guidance` advising how to pursue it. Both are natural language and are passed through to the worker. The blueprint specifies the investigative objective, not literal search strings, leaving the worker free to adapt its phrasing to the claim at hand. A node may declare several actions, which are dispatched together within the same iteration.

Synthesis nodes perform no external retrieval. They issue a model call that consolidates the evidence gathered so far and determines where execution proceeds. These are the decision points of the workflow.

Edges carry an `if` condition and a `to` target, where the target is another node or the terminal `finalize`. Conditions are predominantly natural-language predicates judged by the routing model, e.g. “quote source is missing, context is ambiguous, or counter-evidence has not been searched”, rather than boolean expressions over features. This reflects a deliberate division of labor. Feature predicates govern selection, where the inputs are cheap and reliable and determinism is worth having. Model-judged predicates govern in-run routing instead. There the operative question (“is the attribution conclusively established?”) is a judgment about accumulated evidence that no feature vector could express. An unconditional edge is written as the literal `continue`. Where a node has only one outgoing edge the transition is taken without consulting the model at all.

The verification graph is not executed directly. It is supplied to an LLM planner as strategic guidance rather than as a rigid program. At an action node the declared actions are presented as suggestions, so the planner

may add, omit, or reformulate work in response to what the claim actually requires, and can react to special cases the blueprint’s author did not anticipate. This freedom is bounded at the edges. Routing is confined to the transitions the graph declares, and unconditional transitions bypass the model entirely, so the planner shapes the substance of each step while the blueprint retains control of the skeleton. A blueprint therefore separates *what to investigate and in what order* from *how to investigate*, which is left to the planner. Figure 4.2 depicts a full blueprint for processing claims with media.

4.2. Blueprint Selection

Selection maps an incoming claim to exactly one blueprint. It runs in two stages starting with a deterministic rule filter that establishes which blueprints are *eligible*, followed by a tie-break that chooses among the survivors when more than one remains. The filter is fixed and the tie-break is served by two interchangeable mechanisms, either an LLM call over the candidates’ routing surface, or a learned embedding probe. Both can also be combined, the probe deciding where it is confident and deferring to the LLM elsewhere.

4.2.1. Claim Features

Both stages operate on a typed feature vector extracted from the claim, whose entries fall into two groups that differ in reliability.

Structural features are computed deterministically from the claim object and its text: whether images or video are attached and how many, whether the text contains a URL, a date expression, or a question mark, its length, and whether the claim record carries an attributed author, a stated origin, a publication date, or further metadata. They are cheap and always available.

Semantic features describe what a claim is *about* rather than what it structurally contains: whether it asserts a place or date, asserts an identity, asserts synthetic origin, asserts recontextualization of genuine media, or constitutes a quote attribution, a statistical claim, a scientific or medical claim, or a document screenshot. They are produced by a single cheap LLM pass that judges meaning rather than wording, so that claims in any language are labeled consistently. Because that pass can fail, semantic features are **tri-state** (true, false, or undetermined) rather than boolean, and the extractor is instructed to answer “unknown” whenever the claim does not settle the question. A condition on an undetermined feature is treated as satisfied and therefore never eliminates a blueprint. The distinction being preserved is between *we established this is not a quote attribution* and *we do not know*, only the first of which justifies exclusion. A failed extraction consequently degrades into the tie-break, which reasons over the claim directly, instead of silently misrouting it. The extraction is retried before being abandoned. Needlessly losing the gating signal pushes claims onto the tie-break that the filter could have decided outright.

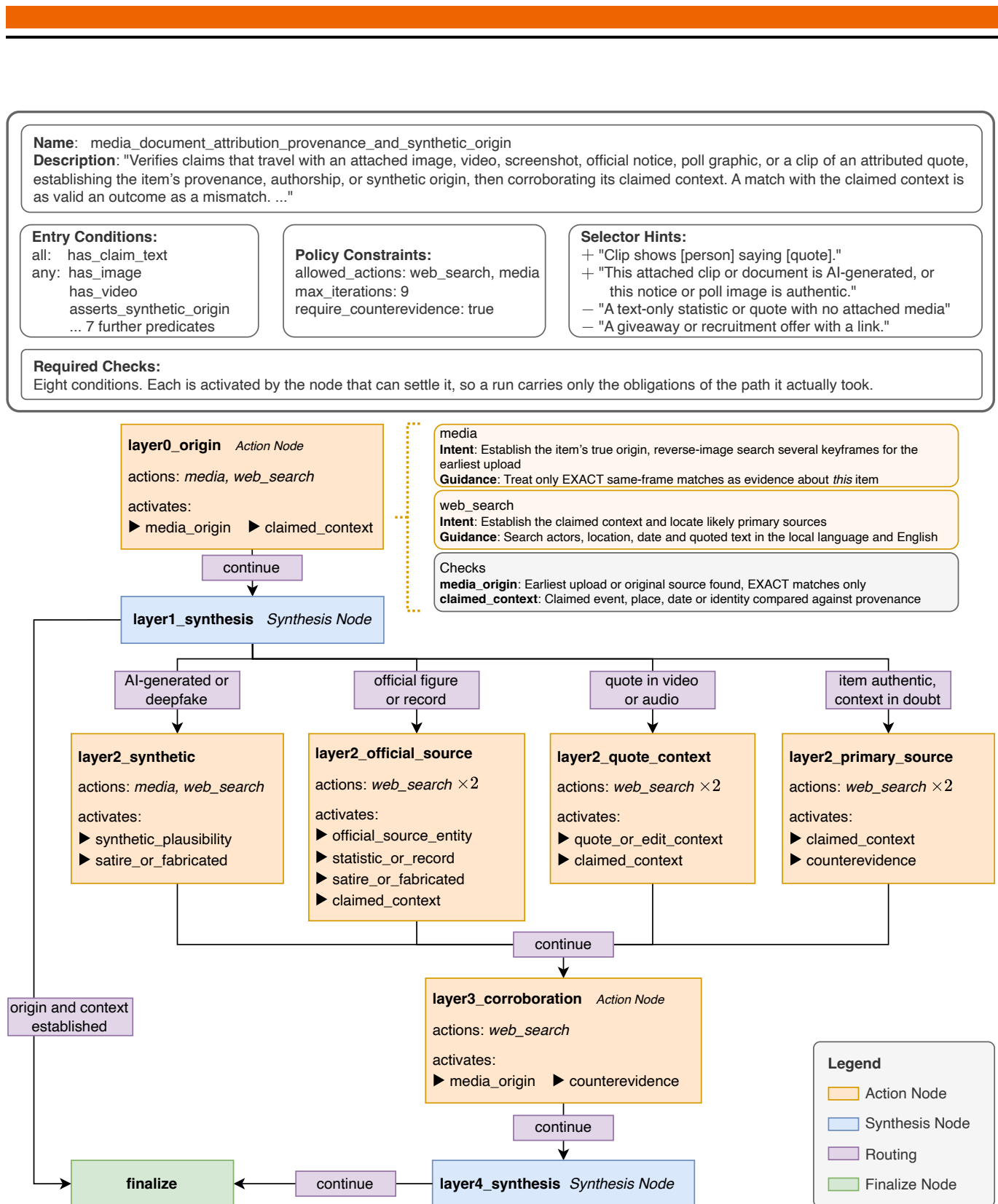


Figure 4.2.: A learned blueprint, shown as the data it is (top) and as the workflow it declares (bottom). This is `media_document_attribution_provenance_and_synthetic_origin` from the pool of Section 4.4. Long field values are abbreviated, check identifiers are shortened, and the natural-language edge conditions are paraphrased. Action nodes dispatch work to worker agents and blue nodes only consolidate evidence and route. One action node is expanded to show that an action carries an intent and query guidance rather than a literal search string.

4.2.2. Stage One: Rule-Based Filtering

Every blueprint except the designated fallback is evaluated against the feature vector by its entry conditions. The outcome determines what happens next:

Exactly one survivor The claim is routed there directly. No model is consulted, and selection is fully deterministic.

No survivors The claim is routed to the fallback blueprint, which declares no entry conditions and is deliberately generic.

Two or more survivors The claim proceeds to the tie-break.

The filter is purely eliminative and it establishes eligibility and never expresses preference. Its value is that eligibility facts which are not judgment calls (e.g. a media-centric workflow requires media to work on) are settled without spending a model call and without any possibility of being overridden downstream.

4.2.3. Stage Two, Mechanism A: LLM Tie-Break

The tie-break presents the claim, its extracted features, and the surviving candidate blueprints to a language model, which returns the name of the blueprint it selects. Each candidate is rendered as its routing surface only (name, description, and the positive and negative features and examples of its selector hints) so the decision is made on what a blueprint advertises about itself, not on its internal strategy.

Two properties of the prompt address failure modes observed in practice. First, the model is told that every candidate has already passed the hard filter, so mere applicability is not a reason to choose one. It must select the *narrowest* blueprint that applies, and treat a broadly-worded candidate as a fallback rather than an answer. Without this, a permissively-worded blueprint absorbs traffic that specialized ones should receive. Second, the model must name a *discriminator* as the concrete property of *this* claim that makes the chosen blueprint fit and the closest runner-up not fit. Being unable to name one is evidence that the candidates were not actually distinguished, which makes the failure visible in the trace instead of silent.

The mechanism is constrained and retried. A selection naming a blueprint that did not survive the filter is rejected rather than honored, so the model cannot overturn a deterministic gate. Responses that are unparseable, truncated, or schema-invalid are retried up to three times with an appended repair instruction, since the relevant failure is not a transport error but a successful call whose body cannot be read. Only after exhausting the retries does the claim fall back to the generic blueprint.

4.2.4. Stage Two, Mechanism B: Embedding Probe

The second mechanism replaces the language-model call with a learned classifier, namely a multinomial logistic regression from the L2-normalized embedding of the claim text to a blueprint. It is available only for learned blueprint pools, where each training claim carries a known target blueprint and the fitted parameters are stored alongside the pool. The fitting procedure is described in Section 4.4. At inference every comparison is claim-to-claim (only the claim is available at run time). The probe therefore learns the routing decision directly from data, whereas the LLM tie-break must infer it from prose descriptions written about that data. It is also substantially cheaper, using only one embedding call and a matrix multiplication in place of a generative call on a prompt containing every candidate. A prediction naming a blueprint the rule filter rejected is never

honored. The probe's ranking is restricted to the surviving candidates instead and the highest-scoring survivor is selected, so the learned component reorders the eligible set without ever widening it. Where no survivor appears in the probe's label set at all, the claim falls through to the tie-break. Classes with too few training claims to cross-validate are dropped at fit time, so the probe simply never routes to them and the tie-break continues to handle those claims.

4.2.5. Combining the Two Mechanisms

The hybrid method routes with the probe where it is confident and defers to the LLM tie-break otherwise, thresholded on the probe's predicted class probability. The arrangement exploits the complementary failure profiles of the two: the probe is accurate and cheap on claims resembling its training distribution, and uninformative on the rest, which is precisely where a model reasoning over descriptions from scratch retains an advantage.

4.3. Agent-Based Blueprint Execution

The system is assembled from four agents behind one shared interface. The following subsections introduce that interface and then each agent in the order a claim passes through them.

4.3.1. Agent Framework

All agents share one interface. An agent is called with a session and returns a result, and is addressed by the orchestrator only through this. Each agent carries a natural-language capability description, which is what the top-level planner reads when deciding whom to delegate to, so adding a worker type does not require changing the orchestration logic.

Session-Scoped State. Agents hold no state across calls and all state lives in the session object passed into the run. A session owns its goal, the claim under investigation, the messages it has seen, and the evidence accepted into it, along with the harness-level constraints described below. Evidence ownership is session-local, so delegation creates a child session seeded with the parent's accumulated evidence, and what the child discovers flows back as a result message that the parent merges into its own state. This keeps every agent independently runnable and testable, and it makes a delegated task resumable, since re-using a session id continues that worker's investigation with its prior context intact rather than starting over.

Failures and Data. A result carries an error list and a terminal status alongside its evidence. Recoverable failures, such as an unreachable page, an unparseable planner response, a hallucinated media reference, or a delegation to an unconfigured agent type, are recorded and the run continues in a degraded form; only unexpected exceptions propagate. The motivation is that in a benchmark run an aborted sub-task turns a single bad retrieval into a missing data point. Partial evidence still reaches the judge, and the error list makes the degradation visible in the trace instead of silent. Every model-driven decision is requested as strict JSON and validated against a typed schema. A response that fails to parse triggers one focused repair call that re-asks for the schema alone before the step is treated as failed. Invalid enum values are coerced to their

non-committal variant with a recorded warning rather than dropped silently. The evaluation harness sets a cutoff date and a set of blocked URLs on the top-level session, and every child session inherits both. The original fact-checking article should already be excluded by date filtering but to prevent leakage due to e.g. malformed dates, the specific URL is always also added to the blacklist. Agents can never widen them. This is the leakage guard for benchmark runs as it prevents a worker from admitting a source published after the claim, or a fact-check article that is itself the answer key.

Traces. Every run is recorded in full. Tracing is hierarchical and mirrors the delegation structure, so the top-level fact-check opens a root scope, each delegated task opens a child scope beneath it, and the worker's own record nests inside that child. Every agent shares one recorder base, which means a new worker type cannot be added without producing a trace. The trace captures the complete decision surface of the run. Every model call is stored with its full prompt and raw response, repair calls included. Alongside those sit the blueprint selection with its surviving candidates and their rejection reasons, every search query with the pages it returned, and the evidence extracted from each page. The record also holds every tool result, the required-check ledger at each step, the node transitions as an explicit graph, errors keyed by phase, and per-model cost, token, and timing accounting. A viewer renders the resulting tree¹. Two things follow. The first is interpretability. A verdict is never a bare label, because the whole path from claim to label can be replayed step by step. A wrong verdict can then be attributed to the stage that produced it, whether a query that was never issued, a page that could not be retrieved, a check that stayed open, or a source the judge overweighted. The second is experimental control. A recorded run can be re-entered rather than re-run, since the session state is reconstructible from its trace, covering the accumulated evidence, the ledger with its reasons, and the delegated-task history. An intervention that changes only the late stages of a run can therefore be evaluated with the entire earlier trajectory held fixed, instead of being buried in resampling noise at temperature 1.0. Section 5.4.1 uses exactly this. The same property lets the judge be re-run over an evidence set recorded earlier, which is why referent status is resolved once at evidence assembly and stamped onto the evidence items, rather than being computed inside the judge. Judging stays a pure function of claim and evidence, and an archived run re-judges to the same statuses.

4.3.2. Fact-Check Agent

The fact-check agent is the top-level controller of a run. It runs no searches, invokes no tools, and assigns no verdict label. Its entire job is to decide what should be investigated next, by whom, and when there is enough to stop, and to keep a record of what has and has not been established along the way. To do this, it selects and uses a blueprint (Section 4.1) as guidance. All actual evidence acquisition is delegated to the web-search (Section 4.3.3) and media (Section 4.3.4) agents. The final label is assigned by the judge (Section 4.3.5) from the evidence set the orchestrator assembled.

Blueprint Handling. The blueprint is treated as guidance, not as a program, and the model is free to delegate tasks other than the node's suggested actions. The graph constrains the shape of the investigation (what stages exist, what must be checked before the next stage, how deep the run may go); the model retains freedom in the content (which queries, which media, how many parallel angles). The reason is that a hard-coded strategy generalizes badly across claim types, while a free-running agent tends to converge on a single generic pattern (search → summarize) and drops the claim-type-specific steps that determine whether a check actually

¹<https://cptk.github.io/mafc-blueprint-learning/index.html> (Note: images are not available here)

```

=== SYSTEM MESSAGE (blueprint + runtime position) =====
<orchestration contract: two phases per iteration; all tasks in one decision run in parallel; follow_up_to resumes a
  session; one media item per media task, addressed by its reference tag>

Available sub-agents: media: <...>, web_search: <...>
Selected blueprint: <name> -- <description>
Policy: allowed_actions=[web_search, media], max_iterations=5,
  require_counterevidence_search=true, max_layer=4

Blueprint graph:                                     # all nodes, every iter.
- node layer1_synthesis (synthesis)
  transitions:
    <origin confirmed, authenticity established, text-media
      consistent, no open questions>          -> finalize
    | <origin unclear, authenticity unresolved, or counter-
      evidence not yet searched>              -> layer2_counter_search
- node layer2_counter_search (actions)
  actions: web_search x3
  activates checks: cross_source_framing_checked
  transitions: continue -> layer2_synthesis
- ...                                             # remaining nodes, ditto

=== USER MESSAGE (claim + node + accumulated state) =====
Required checks:                                     # ACTIVE checks only
- media_origin_investigated: <description>
- text_media_consistent: <description>
- counterevidence_checked: <description>          # ... and others

Current position:
- current node: layer2_counter_search (layer 2), budget left: 3
- stay_allowed: true
- allowed_next_nodes: layer2_synthesis (if: continue)
- unresolved checks: text_media_consistent, counterevidence_checked
- delegated tasks: t1_media(media), t2_origin(web_search)

Claim: "<text with inline media refs, e.g. <image:42>>"
  Author/Date/Origin: <...>
Iteration: N / remaining budget: 3
Claim modalities: images=1 (image:42), videos=0,
  media_delegation_allowed=True
Current node: layer2_counter_search
Suggested actions:
- web_search | intent: <find facts contradicting the claim>
  query_guidance: <alternatives, corrections, debunks>
- web_search | intent: <investigate who published the claim>
  query_guidance: <outlet, account, reliability assessments>
- ...                                             # further node actions

Accepted evidence summaries:                         # takeaways, not raw text
- Source: <url> | Summary: <takeaways, may contain <image:83>>
- Source: <url> | Summary: <...>
Action history: delegate <rationale>; t1_media -> 3 ev., 0 err.
Delegated tasks: t1_media: media, iter=N-2, follow_up_to=None
Required check status:
- media_origin_investigated: supported -- <reason>
- text_media_consistent: unchecked
- counterevidence_checked: unclear -- <reason>

Decide which tasks to delegate, or finalize if evidence suffices.
Return strict JSON only:
{"decision_type":"delegate|finalize","rationale":str,"tasks":
  [{task_id, agent_type, instruction, follow_up_to, rationale}],
  "final_answer":str|null}

```

Figure 4.3.: Planner input at an action node in iteration N ; angle brackets denote abbreviated content. The system message holds only the claim-invariant prefix and the per-iteration state is carried on the user message. Media references resolve to actual image attachments when the message is built. Values are illustrative.

succeeds. Every iteration re-renders the planner's full situation, so it never has to reconstruct its position from conversation history. The rendering is split across two messages, and the split is a caching decision. The system message carries only what stays constant for the lifetime of a claim, namely the orchestration contract, the worker roster, and the blueprint's identity, policy, and graph. Because that prefix is byte-identical across iterations, the provider can serve it from its prompt cache. The volatile state rides at the front of the user message instead, where it cannot invalidate the cached prefix. It covers the check ledger, the current node and layer, the remaining budget, and which transitions are currently available. Figure 4.3 shows a schematic input to the planner at iteration N .

The Iteration Loop. Each iteration executes exactly one node in two separate phases. The blueprint's declared budget bounds how many iterations a run may take, subject to the floor of Section 4.4.6.

1. Execution Phase: At an action node, the planner receives the claim, the node's suggested actions, the accumulated evidence summaries, the delegated-task history and the current check status, and returns a JSON decision to either *delegate* with a list of tasks, or *finalize*. At a synthesis node no decision is asked for and the agent auto-generates a synthesis from the evidence gathered so far and carries it into the routing phase.
2. Routing phase: Where the node offers several outgoing transitions, a second, separate call chooses among them. Where only one transition exists, the agent advances without an LLM call and where none exists, the run ends.

Delegation. Tasks are the unit of delegated work. Each carries a task ID, an agent type (`web_search` or `media`), a free-text instruction, and optionally `follow_up_to` pointing at an earlier task. All tasks emitted in one decision can run in parallel, which makes it cheap to pursue several independent angles in a single iteration. Using follow-up reuses the earlier task's child session id, so the worker resumes with its previous context. Omitting it starts a fresh session. Child sessions inherit the full accumulated evidence set, the cutoff date, and the blocked-URL list from the parent, so worker agents can build on and avoid duplicating prior findings. Returned evidence, messages and errors are merged back into the orchestration state. The planner sees the worker's distilled takeaways rather than raw scraped page text, so retrieval boilerplate does not crowd out the strategic state; the full evidence, including items that yielded no takeaways, still reaches the final synthesis and the judge.

Required Checks. Required checks are the mechanism by which a blueprint states what must be established regardless of the path taken (e.g. the origin and earliest appearance of the media was investigated, counter-evidence was explicitly searched for). Each is tracked as `unchecked` / `supported` / `refuted` / `unclear` with a reason. Checks activate lazily, which means that blueprint-level checks are active from the start, while checks attached to nodes activate only when execution first reaches that node. A claim therefore only carries the obligations of the path it actually took. This is especially important for multi-lane blueprints, where an unvisited lane's checks would otherwise reach the verdict as unresolved uncertainty and penalize the run for work it had no reason to do. Updates normally ride along with the routing decision. Because routing makes no LLM call on single-exit and terminal nodes, a separate, focused check-update call fills that gap. It deliberately omits the blueprint system prompt (no routing choice is being made) and is skipped whenever it could learn nothing, i.e. when there are no open checks, no evidence, or nothing changed since the last evaluation. Without this, a check activated on a long single-exit chain would stay unchecked for the rest of the run and arrive at the verdict as unresolved uncertainty.

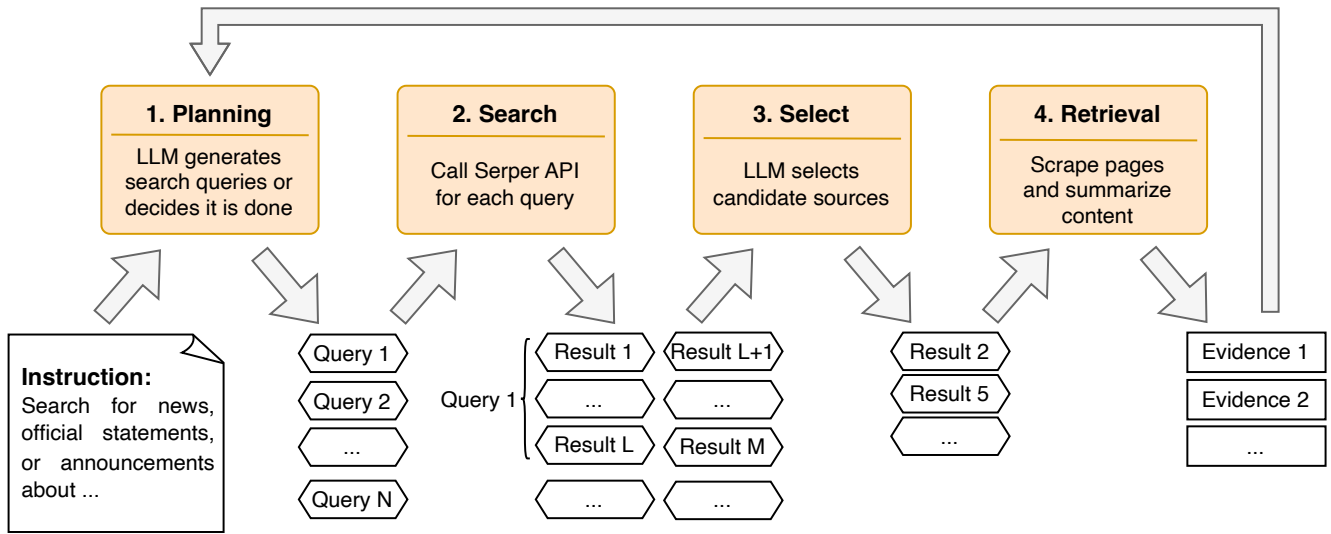


Figure 4.4.: Pipeline for the web agent. Given an instruction from the fact-checker, an LLM call plans the round, either issuing a set of search queries or declaring the task complete. The queries are executed against the search API under the claim’s date cutoff, and the returned candidates (URL, title, snippet, date) are passed to a second LLM call that selects the relevant ones across all queries at once. Selected pages are retrieved and summarized into evidence items (summarized text plus media). The evidence accumulated so far is carried into the next round’s planning call.

Termination. A run terminates normally in one of four ways: (1) the planner finalizes at an action node, (2) routing selects finalize, (3) the current node has no outgoing transitions, or (4) the iteration budget is exhausted. In every case where evidence is gathered, if no final answer has been produced, the agent synthesizes one from the accepted evidence and the check ledger, with unresolved checks explicitly rendered, so what was not established is carried into the verdict rather than silently dropped. Before evidence is handed to the judge, the agent collapses repeated findings and stamps referent status across the assembled set. Repetition is judged per finding rather than per source, since several distinct results can concern the same media item.

4.3.3. Web-Search Agent

The web-search agent handles tasks requiring information retrieval from the open web. It receives a free-text instruction from the orchestrator, together with the claim under verification and a cutoff date, and operates in a loop so that each round’s findings inform subsequent searches. Figure 4.4 gives an overview.

Each iteration proceeds in four steps. The iteration budget, the number of queries per step, and the number of results per query are configurable.

Plan An LLM call receives the task and the prior context (evidence accepted so far, rendered as source/summary blocks) and returns a set of queries together with a completion flag. The prompt states that multiple iterations are available, so that the model builds on earlier findings rather than front-loading all queries into the first round. Malformed output triggers a repair call; if repair also fails, the raw task text is used as a single query in the first iteration, and the run terminates in later ones.

Search Queries are dispatched concurrently, each with a result limit and an upper bound on publication date derived from the cutoff. Query strings issued in earlier iterations are discarded. A query that returns nothing is retried once with its quotation marks and search operators stripped. The retry is spent only where the first attempt already produced no sources, so it can never displace a productive query. The motivation is measured. Across a run of one thousand claims, 69.8% of quoted queries returned zero sources against 12.6% of unquoted ones, and the gap held both within individual claims (69.8% against 14.7%) and within each script, which rules out difficulty and language as the cause.

Select Above a threshold of five candidate URLs, a second LLM call applies a relevance filter across the results of all queries at once rather than filtering each query’s results independently. One judgment is therefore made with the whole candidate pool in view, so a result can be discarded on the evidence that stronger candidates exist elsewhere in the round, which per-query filtering cannot see. Per-query attribution is retained, so each selected URL remains associated with the query that surfaced it. If the filter call fails, all candidates are passed through. The failure mode is over-inclusion rather than silent evidence loss.

Retrieve & extract Selected URLs are fetched, skipping those already retrieved in earlier iterations. One LLM call per page then extracts facts under strict constraints: only explicitly stated facts, no inference, no verdicts, exact figures, dates, attributions, and their qualifying conditions preserved. A page marked as containing no relevant content yields no evidence. Relevance is anchored to the claim rather than the query, following evidence that retrieval which optimizes topical overlap with the query leaves the retrieved set poorly aligned with the item under verification [97]. Admitting such pages fills the fact budget with background that does not bear on the claim, and verification accuracy declines as these items accumulate. Images and videos extracted from pages are carried alongside the textual evidence, filtered for relevance by the same calls, and deduplicated. Where a page cannot be retrieved at all, the search-engine snippet is retained but explicitly labeled as incomplete, so that a gist-level snippet is not mistaken downstream for a verified full read.

The loop terminates when the planner signals completion, when it returns no usable queries, when an iteration produces no retrievable results, or when the iteration budget is exhausted. A final LLM call then synthesizes across all accepted evidence, with instructions to state agreements and disagreements between sources and to identify missing evidence. If this call fails, the evidence blocks are concatenated and returned unsynthesized.

Details on Web-Search Queries issued by the planner are passed to the Serper API² together with the cutoff date, which Serper applies as a date-range restriction. The parsed response yields an ordered list of (URL, title, release date, preview snippet) tuples. The returned URLs are then used to retrieve the actual page contents. A plain HTTP request with text extraction is attempted first. It is treated as unsuccessful not only on network or status errors, but also when the extracted text falls below a minimum length, which serves as a heuristic for JavaScript-only shells and login or anti-bot walls. In that case retrieval falls back to scrapeMM³, which employs more sophisticated mechanisms including Firecrawl⁴ and Decodo⁵.

²<https://google.serper.dev>

³<https://github.com/multimodal-ai-lab/scrapeMM>

⁴<https://github.com/firecrawl/firecrawl>

⁵<https://decodo.com/>

4.3.4. Media Agent

The media agent handles tasks requiring the analysis of media items (images or videos) attached to a claim. It accepts a free-text instruction alongside a media item and follows a single-pass workflow.

The agent begins with a planning step in which an LLM selects which of the available tools, described below, to invoke for the given instruction. Any subset may be selected. Two fallbacks cover the cases where that selection yields nothing usable. An unparseable planner response falls back to reverse image search and geolocation, the two tools that apply to any media item. A parseable response naming no tool at all falls back to reverse image search alone, since establishing provenance is the minimum useful check for a media claim and analysis without any tool produces nothing. Finally, an LLM call synthesizes the evidence into an answer for the caller, emphasizing what is relevant to the given task. The same call names which evidence items the answer rests on, and only those are returned upward, so the step acts as a filter as well as a summary.

Reverse Image Search Tool (RIS)

The goal of the RIS tool is to find places where the exact same media item has been published. It submits the media item to the Google Cloud Vision web detection API⁶, which returns a list of web sources where matching or visually similar images have appeared, along with detected visual entities and topic labels. This API does not natively support videos. A single frame is often a black or title intro frame, which is a weak basis for finding a video's origin. For this reason, in these cases 4 frames are sampled evenly, always including the first and the last frame, and each is passed to the API. The results of individual frames are then aggregated by the following rules: (1) deduplicate URLs, keeping the order in which they were first seen, (2) entities get the max score over all frames, and (3) topic labels are unioned. The merged entity list is finally sorted by descending score.

The returned pages are then post-processed. They are first ordered such that pages containing a full image match precede all others, after which at most one page per base domain is retained (ignoring the `www.` and `m.` subdomain prefixes). Without this restriction, a single item that circulated widely on one platform would fill the entire result list with near-identical social media posts and crowd out independent sources; ordering full matches first ensures that the page surviving the filter for a given domain is the one with the strongest match.

Every retained page is annotated with the *match precision* that the API reports for it. A page is labeled `EXACT` if the API confirms that the identical image appears on it, `PARTIAL` if only a cropped, edited, or overlapping version was matched, and `unknown` if the page was returned without either confirmation. This distinction is the central output of the tool because an exact match establishes that a statement made on that page refers to *this* media item, whereas a partial match may well describe a different image of the same event, and an unconfirmed page says nothing about the item at all. Accordingly, only pages with a confirmed exact or partial match are promoted into individual evidence items; visually similar pages are discarded, since treating them as findings would attribute claims about other media to the item under investigation. In addition, up to two direct URLs of the matched image files themselves are recorded per page. These are the most precise referent pointers available, as they can be retrieved directly for a frame-level comparison against the claim's media without having to scrape the surrounding page, which is frequently protected against automated access.

The absence of any match is treated as a finding rather than as an empty result. If no page with a confirmed match is found, the tool reports explicitly that no provenance could be established, i.e. that the origin of the media item, its earliest appearance, and its link to any claimed event remain unverified. This negative result

⁶<https://cloud.google.com/vision>

is propagated to the planner and to the verdict stage in the same way as a positive one, because the failure to locate a widely reported image is itself diagnostic. The detected entities and topic labels, in contrast, are only reported as coarse context for the item as a whole and are deliberately not attached to individual pages. They describe the neighborhood of the query image in the search index rather than the item itself, and were observed being quoted downstream as if they were content-level evidence.

Geo-Location Tool

The geo-location tool determines where an image or video was recorded, which allows a claimed location to be checked against the visual content itself. Two specialized geo-location models [49, 20] were evaluated against the application of general-purpose LLMs to the same task. Appendix A.3.1 describes the analysis setup and reports the full results. General-purpose LLMs surpass both specialized models in country-level accuracy by a large margin, which is why GEMINI 3 FLASH is used for this task. The tool is given the image, or a single sampled frame for videos, and returns its three most likely countries ordered by plausibility. The prompt directs the model to judge only from visible cues such as architecture, vegetation, terrain, vehicles, signage and script, and clothing. It is asked for the ranking alone, without an accompanying rationale, so the output stays a compact signal rather than prose the downstream synthesis would have to weigh.

Using an LLM has a further advantage over the specialized models beyond the accuracy difference. Those models are closed-set classifiers over a fixed list of candidate countries and always produce a ranking, so an image without any geolocatable content, such as an indoor close-up or a screenshot, still yields a confidently ranked list of countries. The LLM can instead abstain and state that no guess is possible, which turns an unanswerable query into an explicit negative finding rather than a spurious location. What the LLM reports remains a self-assessment rather than a calibrated probability, so the ranking is treated by the downstream stages as a soft indicator rather than as decisive evidence.

Manipulation and AI Detection Tools

Many images and videos in fact-checked claims are either fully AI-generated or contain manipulated elements such as swapped faces, so a detector for file-level integrity (as opposed to misleading use of an authentic file) would be a valuable signal. Four tools were implemented and evaluated on the images of the VERITAS 2026 Q1 split: (1) the C2PA provenance checker, (2) TRUFOR [47], (3) GEND [164], and (4) the commercial Sightengine⁷ API. While C2PA can natively handle videos, the other three use frame sampling for analysis. Appendix A.3.2 gives more details on the methods and analyzes the performance of the different detectors. C2PA manifests do not survive redistribution and are absent from every image in the corpus, TRUFOR is barely above chance while flagging 79% of authentic files, and GEND scores below chance. Only Sightengine’s AI-generation head is usable, and only as a one-sided signal. A positive detection can be trusted, but silence establishes nothing. In Section 5.4.2, we compare the pipeline with no detector, with Sightengine, and with a ground-truth oracle that bounds what *any* detector could contribute. The oracle produces no significant improvement on the primary metric, so manipulation detection is implemented but disabled by default.

⁷<https://sightengine.com/>

4.3.5. Judge Agent

The judge is the terminal stage of a fact-check pipeline. It does not investigate on its own and everything upstream (blueprint selection, delegated web-search and media tasks, iteration, synthesis) exists to produce an evidence set; the judge’s only job is to turn that evidence set plus the claim into exactly one benchmark label with a short justification. While the evidence retrieval process is benchmark-agnostic, the judge is the only component that knows the benchmark’s label space, its class definitions, and its scoring conventions. Unlike the other agents, it uses no external tools and follows a simple, single-pass workflow.

The judge prompts a model M_{judge} with a prompt that is composed of four components:

Claim text, author, posting date, origin. Because the judge read the claim date as an event date in preliminary experiments, it also gets the disclaimer that date refers to the date of the claim being shared.

Allowed labels with their definitions as the closed vocabulary it must pick from.

The media referent block (for claims with media items only), see below

Deduplicated accepted evidence contains one block per source with URL + the summarizer’s takeaways, each optionally annotated with a referent tag.

The system message is deliberately minimal and constraining: predict exactly one label, use only the provided evidence, prefer the uncertainty label when evidence is thin. Everything benchmark-specific is appended as extra judge rules. For VERITAS these rules conceptually do three things:

Calibrate confidence. The 7-class scheme forces the judge to commit to a direction (🟢 **Intact** / 🟡 **Compromised**) and a strength (certain / rather certain / rather uncertain). Because these labels have been shown to be heavily influenced by the LLM’s calibration, the rules define hard anchors. “certain” requires a direct primary source and no residual doubt; a conclusion resting on multiple corroborating secondary sources is capped at “rather certain.” The default when in doubt is explicitly downward.

Block the wrong-referent flip. The dominant residual error is a debunk that in fact applies to a different video of the same event. The rules state that a debunk bears on the claim only where the media is confirmed to be the same, and that “similar videos were debunked” never on its own justifies compromised.

Block the mirror-image error. For “this video shows X” claims, proving X happened says nothing about whether this footage depicts it. The rules impose a ceiling that without positive provenance for the media itself (original upload, exact reverse-image match, explicit authentication), the label cannot exceed “🟢 **Intact** (rather uncertain).”

The remaining rules address recurring surface errors including superlatives the sources do not state, statements later denied, and nominal imprecision in secondary details.

Media referent linkage. The last two rules above turn on one question the evidence does not answer by itself. When a source discusses “this video”, is it discussing the video the claim carries? The media a source actually refers to is its *referent*. A prominent event attracts many recordings, some genuine and some fabricated, so establishing the referent is what decides whether a source speaks to this claim at all. Reverse image search answers the question for part of the evidence, because it reports whether a page carries an exact copy of the claim’s media or only a similar one. That report arrives as one evidence item, while the assertion it qualifies typically sits in a different item, retrieved from the same page by text search. Nothing joins the two. We therefore resolve a referent status for every source once the evidence set is assembled, marking it *confirmed*,

partial, or *unverified*, and attach it to the source it describes. One rule then binds the judge. A statement about the media’s origin, date, context, or authenticity applies to the claim’s media only where its source is confirmed to show that media. The rule holds equally for debunks and for authentications, so it supplies a missing distinction without introducing a directional bias. An unverified source counts as uninformative rather than contradictory, since a missing confirmation is not evidence of different media. Sources the search index leaves unlabeled are checked directly, by comparing the imagery on the page against the claim’s own frames.⁸ What is tested is whether the two are the same image, rather than whether they show the same thing, since two recordings of one event show the same thing by construction. The check is bounded to a handful of sources per claim and takes partially-matched pages first, where a confirmation is most likely to be recoverable. Only positive results are recorded. A page whose imagery does not match may simply not carry the relevant still, and reading that as evidence of different media would reintroduce the original error in mirror image.

Sampling and Aggregation. The judge can be configured to draw several independent verdicts from the same prompt and aggregate them. Where the benchmark assigns a numeric value to each label, as VERITAS does, the sampled values are averaged; the verdict is that mean snapped to the nearest allowed label, while the mean itself is retained and scored directly by the regression metrics, so a split panel is not charged with the full value of whichever label happened to lie nearest. Where no numeric mapping is available, a majority vote decides. The justification reported is taken from a sample whose label agrees with the verdict, so the recorded rationale supports the recorded label. Under numeric aggregation the snapped label is occasionally one that no individual sample chose, and the justification of the first sample is reported instead.

4.4. Blueprint Learning

Sections 4.1 and 4.2 treat the blueprint pool as given. This section produces one, deriving it from a corpus of professional fact-checks with no manual curation at any stage. Figure 4.5 gives the shape of the pipeline, and the subsections that follow take one stage each. A pattern recurs throughout. Generative stages produce blueprints that are individually plausible and jointly inconsistent, so each is followed by a deterministic pass that repairs what the model got wrong.

4.4.1. Notation

Let $\mathcal{D} = \{(c_i, a_i)\}_{i=1}^N$ be the training corpus, where c_i is a claim and a_i the fact-checking article that adjudicated it. Write $\mathcal{B}$ for a blueprint pool, $\iota(B)$ for the iteration budget of blueprint B , and $L(B)$ for the length of the longest path through its verification graph.

An LLM extractor M_{analyze} maps each article to a structured analysis $A_i = M_{\text{analyze}}(a_i)$. This records the claim type, a short verdict summary, the concrete evidence the article cites, the types of evidence used, an ordered sequence of investigative actions each linked to its finding and flagged as decisive or not, the high-level investigative steps, any search queries the article names, and a process-richness rating $r_i \in \{\text{full}, \text{partial}, \text{result_only}\}$. Richness records how much of the investigation the article actually describes,

⁸The comparison uses perceptual hashing by default and enables crop-tolerant keypoint matching only where the index has already reported a partial match on the page. Ungated, keypoint matching confuses different frames of one event that share a broadcaster’s on-screen banner, which is the very confusion this mechanism exists to remove.

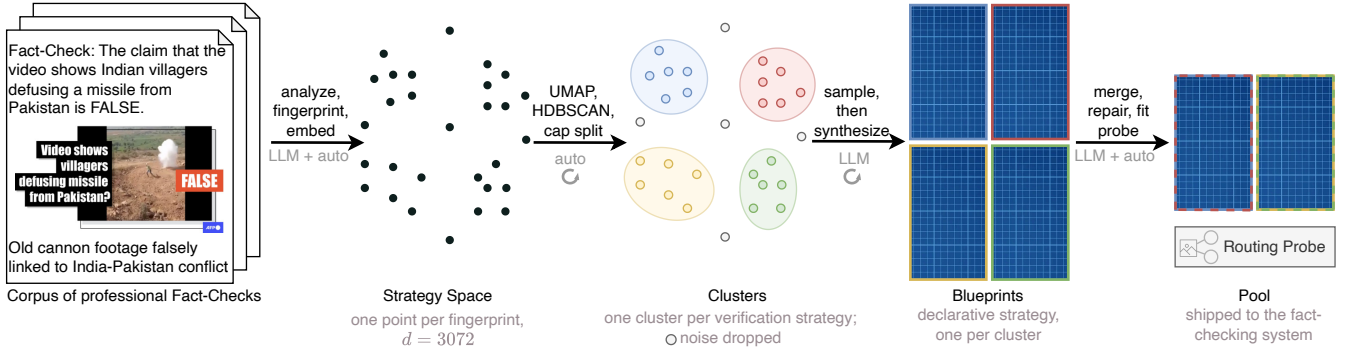


Figure 4.5.: Offline construction of the blueprint pool. Each professional fact-check is reduced by an LLM to a structured record of how the check was carried out, stripped of topic vocabulary, and embedded, which places one point per claim in strategy space. Density-based clustering groups claims resolved by the same method regardless of subject, discards points belonging to no coherent family, and recursively splits any cluster that would exceed a share τ of the corpus. Representatives spanning each cluster’s core and its spread are passed to a synthesizer that writes one blueprint per cluster, which may itself ask for a further split. A final pass merges blueprints differing only by topic and repairs the inconsistencies a language model reliably generates, among them under-declared iteration budgets and entry conditions that exclude a blueprint from its own claims. It also fits the routing probe, which is what allows a pool built from articles to be applied to claims alone. Arrow labels state which stages are model-driven and which are deterministic. Colors track each cluster through to the pool, where two merges have taken place.

separating articles that report only a verdict from those that narrate a method. Where an article is silent about process, the action and step fields are left undefined rather than empty, so that *not described* stays distinguishable from *described as empty* (the same refusal to encode absence as negation that governs the tri-state semantic features of Section 4.2).

Only articles carrying process signal are usable. Write $\mathcal{D}'$ for the retained corpus,

$$\mathcal{D}' = \{ (c_i, a_i) \in \mathcal{D} : r_i \in \{\text{full, partial}\} \}.$$

4.4.2. Strategy Fingerprints

The central representational decision is what to embed. Embedding claim text would cluster by *topic* (e.g. elections, vaccines, a particular conflict) whereas blueprints must be distinguished by *method*. A strategy fingerprint $\phi(A_i)$ is therefore assembled from the analysis with topic vocabulary largely suppressed. It carries the claim type, the decisive actions, the evidence types with decisive ones ordered first, the full action sequence, the leading investigative steps, the process richness, and the claim’s modality flags. All fields except the steps draw on closed vocabularies and contain no topical terms. The steps are free text and retain residual topic detail, limited by keeping at most the first five and truncating each to 120 characters. Two claims about unrelated subjects that were both resolved by reverse image search and geolocation produce near-identical fingerprints; two claims about the same election resolved by different methods do not.

Each fingerprint is embedded by an embedding model E and L_2 -normalized,

$$e_i = E(\phi(A_i)) \in \mathbb{R}^d, \quad \hat{e}_i = \frac{e_i}{\|e_i\|_2},$$

with $d = 3072$ for the embedding model used here⁹. The resulting space is called *strategy space*. Note that it is constructible only at training time because ϕ reads the ground-truth article, which at inference does not exist. This asymmetry is what Section 4.2 resolves, see Subsection 4.2.4.

4.4.3. Clustering

The strategy space is clustered by HDBSCAN [17, 87], optionally after a UMAP [88] projection to d' dimensions. Without the projection, density estimation degrades in 3072 dimensions and most points are labeled noise. The result is a partition

$$\mathcal{P} = \{P_1, \dots, P_K\}, \quad P_k \subseteq \mathcal{D}', \quad P_k \cap P_l = \emptyset \quad (k \neq l),$$

together with a noise set $\mathcal{N} = \mathcal{D}' \setminus \bigcup_k P_k$ that is discarded. Density-based clustering is used precisely so that claims not belonging to a coherent strategy family are excluded rather than forced into one. Write $n = |\mathcal{D}'| - |\mathcal{N}|$ for the number of clustered claims, which is the denominator for every share quoted below.

Clusters exceeding a fraction τ of n are then recursively re-clustered,

$$|P_k| > \tau n \implies P_k \rightarrow \{P_k^{(1)}, \dots, P_k^{(j)}\}, \quad \tau = 0.2,$$

first with HDBSCAN in leaf mode on the subset alone, falling back to k -means with $k = \max(2, \lceil |P_k|/\tau n \rceil)$ when that yields fewer than two subclusters. Points labeled noise by the sub-clustering are reassigned to the nearest subcluster by centroid distance rather than discarded, since they were validly clustered one level up. The threshold stays absolute through the recursion (subclusters are measured against n , not against their parent) because what ultimately matters is the share of traffic a blueprint will serve, not its size relative to whatever it was split from. The recursion is bounded at depth three, so the cap is a best-effort constraint rather than a guaranteed invariant. A cluster whose subdivisions remain unbalanced can still terminate above τn . Splitting on the cap alone over-fragments. Leaf-mode sub-clustering shatters an oversized cluster into many small shards, and shards sharing a dominant strategy would yield near-identical blueprints that consolidation must then undo. A repack pass therefore follows the split. Shards descended from the same parent that carry the same strategy label are greedily recombined, subject to the recombined group staying below τn ; shards with distinct labels stay separate. Cap and repack together target a partition whose parts are large enough to characterize one strategy and small enough not to mix several. The cap exists to protect the *synthesis* stage rather than to bound blueprint size as such. A cluster spanning a large fraction of the corpus necessarily mixes several verification strategies, and a synthesizer shown representatives drawn from all of them can only describe the result in terms broad enough to cover them all. Splitting the input is cheaper than detecting the consequence several stages downstream, at routing.

4.4.4. Per-Cluster Synthesis

Each cluster yields one blueprint. Because a synthesis prompt cannot hold an entire cluster, m representatives are selected by a procedure covering both the cluster's core and its spread. Let $\hat{\mu}_k$ denote the normalized cluster centroid,

$$\hat{\mu}_k = \frac{\mu_k}{\|\mu_k\|_2}, \quad \mu_k = \frac{1}{|P_k|} \sum_{i \in P_k} \hat{e}_i.$$

⁹OpenAI text-embedding-3-large

The selection S is initialized with the $\max(1, \lfloor m/2 \rfloor)$ points nearest the centroid,

$$S = \arg \min_{|S'| = \max(1, \lfloor m/2 \rfloor)} \sum_{i \in S'} \|\hat{e}_i - \hat{\mu}_k\|_2,$$

and extended by greedy farthest-point sampling until $|S| = m$, each step adding the point whose nearest already-selected neighbor is farthest away,

$$i^* = \arg \max_{i \in P_k \setminus S} \min_{j \in S} \|\hat{e}_i - \hat{e}_j\|_2, \quad S \leftarrow S \cup \{i^*\}.$$

The representatives are $R_k = \{c_i : i \in S\}$, or the whole cluster when $|P_k| \leq m$. Sampling near the centroid alone would hide heterogeneity. A model shown only core claims cannot detect that a cluster mixes several verification strategies, and would silently average them into one blueprint. Uniform random sampling is the natural alternative and is rejected for the same reason. At the small m a synthesis prompt allows, a random draw represents neither end reliably. It can miss the dense core, so the blueprint is not anchored on the typical claim, and it can miss the rare variants, so a mixed cluster still reads as homogeneous. Drawing the core first and then the spread guarantees the synthesizer sees both the prototype it should center on and the extremes it must either cover or flag for a split.

Synthesis itself is an LLM call M_{synth} that takes the generic blueprint B_0 as a structural template together with the representatives R_k and the cluster’s share of clustered training claims, $w_k = |P_k|/n$, and returns

$$M_{\text{synth}}(B_0, R_k, w_k) = (B_k, \text{split}_k, \rho_k),$$

a blueprint, a split flag, and a rationale. The template supplies the schema, so the model shapes strategy rather than inventing structure. A cluster serving a large fraction of traffic is instructed to include a corroboration layer and to size its investigation accordingly, so the share shapes the depth of the generated graph. The share is also enforced mechanically by the traffic floor of Section 4.4.6, so a blueprint cannot end up below the budget its share warrants even when the model ignores the instruction.

The split flag makes representative sampling actionable. If the model reports that the representatives reveal materially different strategies, the cluster is bisected by 2-means in strategy space and each half is synthesized independently, to a maximum recursion depth of one. Claims lacking embeddings stay with the larger half; if bisection is infeasible, the unsplit blueprint is kept. Without this, the only recourse to a heterogeneous cluster would be to re-tune the clustering hyperparameters globally, which trades a local correction for a change affecting every other cluster.

4.4.5. Consolidation

Synthesis is per-cluster and therefore blind to the pool. Two clusters separated by the density estimator may nonetheless warrant the same strategy. A single LLM call over the whole pool proposes merge candidates, subject to an explicit criterion, i.e. that blueprints are distinguished by *how* they verify, never by *what* their claims are about. Two blueprints following the same investigative structure and differing mainly in subject domain (a country, a religion, a named person) must be merged, since topic-specific routing adds selector noise with no strategic gain; blueprints serving different strategies must not be, and ambiguity resolves toward not merging.

A merge does not combine two graphs. The pair is *re-synthesized*. The base blueprint is passed to the same synthesis operator used in the previous stage, this time with the union of both claim sets as evidence and

an instruction to cover all of them (broadening entry conditions, combining required checks, and extending the verification graph as needed) while staying narrower than the fallback. The second blueprint therefore contributes its claims rather than its structure, and the result is a new blueprint written from the base rather than a composition of the two.

Detected pairs are folded by connected component rather than pairwise. When the detector judges $A \sim B$, $A \sim C$ and $A \sim D$ it has identified one redundant family, not three coincidences; executing the pairs independently would collapse the family by a single merge and abandon the rest. Each component is therefore folded in sequence, with the result of each merge becoming the base of the next, so merges cascade and a blueprint may absorb several others in one pass.

Consolidation is deliberately exempt from the cluster cap of the previous subsection, and in practice it, rather than the clustering, determines the final size distribution. The asymmetry is intentional because splitting an oversized cluster prevents a catch-all from forming, whereas refusing a merge *preserves a distinction that was never real*.

4.4.6. Mechanical Guards

The remaining stages are repairs applied to generated output. They exist because a language model reliably produces blueprints that are individually plausible yet internally or jointly inconsistent.

Path budget One agent iteration executes exactly one node of the verification graph, synthesis nodes included, since each issues its own model call. The iteration budget must therefore be at least the length of the longest path through the graph,

$$\iota^*(B) = \max\{\iota(B), L(B)\},$$

where $\iota(B)$ is the declared budget and $L(B)$ the longest simple path, counted in nodes. Below this bound the deepest branch is unreachable. The run exhausts its budget mid-path and finalizes early, so any checks scoped to that branch never activate and the blueprint appears to run correctly while skipping the conditions it exists to enforce. The bound is a floor and never a ceiling, so a declared budget above it is left untouched and early-exit transitions still end runs sooner. It is enforced whenever a blueprint is loaded rather than only when one is generated, so it holds uniformly for hand-authored and synthesized pools.

Traffic floor The iteration budget is also tied to the share of traffic a blueprint is expected to serve, independently of its graph. A blueprint whose cluster holds at least a tenth of the clustered training claims is raised to at least four iterations, applied before the path budget so the two floors compose to their maximum. The motivation is an observed regression, in which the blueprint absorbing 47% of routed claims ran with three iterations while blueprints serving twenty-nine claims each ran with four. Allocating budget inversely to traffic is precisely backwards, since the claims a blueprint sees most often are the ones whose outcome its budget decides.

Contrast pass One LLM call over the whole pool rewrites each blueprint’s description and selector hints so that the pool partitions the claim space. Each description must state which claims it handles and what distinguishes it from its nearest sibling, and positive hint examples must not plausibly match another blueprint. This serves the requirement established in Section 4.2 where the tie-break reads only these two fields, so routing quality is bounded by how crisply they are worded, and per-cluster synthesis has no view of the pool from which to word them. Revisions that arrive structurally invalid, or that fail a

lint forbidding descriptions which presuppose a verdict, are discarded per blueprint rather than applied pool-wide.

Entry-gate validation A generated entry condition can exclude a blueprint from the very cluster it was built for (e.g. a statistics blueprint gated on `asserts_synthetic_origin`). Such a gate is strictly worse than no gate at all because the blueprint becomes unreachable by its own traffic, and because the filter is eliminative the tie-break never gets the opportunity to rescue it. Self-coverage is the fraction of a sample S of the blueprint’s own claims that its entry conditions still admit,

$$\text{cov}(B, S) = \frac{|\{c \in S : \text{entry}(B) \text{ admits } f(c)\}|}{|S|},$$

where $f(c)$ is the feature vector of claim c . Conditions are dropped greedily, worst-covering first, until $\text{cov}(B, S) \geq 0.7$. The pipeline additionally warns when a blueprint declares no semantic entry condition at all, since such a blueprint survives the filter for nearly every claim and pushes the whole routing decision onto the tie-break, and one the filter cannot detect on its own.

4.4.7. Routing Probe

Clusters live in strategy space, which is built from ground-truth articles; routing must happen in claim space, where only c_i is available. The probe of Section 4.2 is exactly the learned bridge between the two. Every training claim inherits a target blueprint from its cluster,

$$y_i = B_{k(i)}, \quad i \in P_{k(i)},$$

and a multinomial logistic regression is fitted from the normalized embedding of the *claim text* to that label,

$$\pi_\theta(B \mid \hat{e}_i^{\text{claim}}) = \text{softmax}(W\hat{e}_i^{\text{claim}} + b).$$

Every comparison is claim-to-claim; article embeddings never appear at inference. Blueprints with fewer than $2F$ training claims (F the number of cross-validation folds) are dropped from the label set, so the probe simply never routes to them and the tie-break continues to serve those claims. Routing accuracy is estimated out-of-fold under stratified F -fold cross-validation and reported against the majority-class baseline, and the fitted coefficients are written as plain JSON beside the pool.

4.5. Single-File Strategy Baseline

The blueprint system represents procedural knowledge as a routed pool of structured verification plans. To isolate the contribution of that structure, we implement a baseline that keeps every other component fixed and replaces the pool with one free-text document, `Strategy.md`. The document is distilled from the same corpus of professional fact-check articles that supplies the blueprint learner, and it is the only fact-checking guidance the agent receives at inference time. Two stages define the baseline: (1) an offline construction stage that folds the corpus into the document, and (2) an inference stage in which a standalone agent verifies claims under the document alone.

Algorithm 1 Construction of Strategy.md

Require: records $\mathcal{R}$, batch size b , epochs E , seed s , word target W , consolidation period c

```
1:  $D \leftarrow \emptyset$  ▷ or the recommended skeleton
2:  $t \leftarrow 0$  ▷ global fold counter, persists across epochs
3:  $lastCons \leftarrow \text{FALSE}$ 
4: for  $e = 0, \dots, E - 1$  do
5:    $\mathcal{B} \leftarrow \text{BATCHES}(\text{SHUFFLE}(\mathcal{R}, s + e), b)$ 
6:   for  $k = 1, \dots, |\mathcal{B}|$  do
7:      $D', ok \leftarrow \text{FOLD}(D, \mathcal{B}_k, W)$ 
8:      $D \leftarrow D'$  ▷  $D' = D$  when  $ok$  is false
9:      $t \leftarrow t + 1$  ▷ counts every fold, failed ones included
10:     $lastCons \leftarrow \text{FALSE}$ 
11:    if  $c > 0$  and  $t \bmod c = 0$  then
12:       $D \leftarrow \text{CONSOLIDATE}(D, W)$ ;  $lastCons \leftarrow \text{TRUE}$ 
13:    end if
14:     $\text{CHECKPOINT}(D, \text{run state})$ 
15:  end for
16:   $\text{SNAPSHOT}(D, \text{epoch } e)$ 
17: end for
18: if  $c > 0$  and not  $lastCons$  then
19:    $D \leftarrow \text{CONSOLIDATE}(D, W)$ 
20: end if
21: return  $D$ 
```

4.5.1. Construction: Sequential Distillation

The fold recurrence. Records are filtered exactly as in Section 4.4, keeping only articles whose *process richness* is full or partial. Each retained record is presented to the synthesizer as its article analysis, covering the claim, the evidence and steps the checkers used, and how the case resolved. The topic-stripped fingerprint $\phi(A_i)$ is deliberately not used here. Its purpose is to make claims comparable by method for the sake of clustering, and a free-text playbook is never clustered, so suppressing topic vocabulary would only withhold context the synthesizer can generalize from. Let $\mathcal{R} = \{r_1, \dots, r_N\}$ be the retained records and let M_{synth} denote one call to the synthesizer model. For epoch e we shuffle $\mathcal{R}$ with seed $s + e$ and partition it into batches $B_1, \dots, B_{\lceil N/b \rceil}$ of size b . The document evolves by a single sequential recurrence,

$$D_0 = \emptyset \quad (\text{or the seed skeleton}), \quad D_k = M_{\text{synth}}(D_{k-1}, B_k), \quad (4.1)$$

with no merge or reduction step. One LLM call consumes one batch, so a pass over the corpus costs $\lceil N/b \rceil$ calls. The synthesizer is stateless across calls, and the driver owns the running document and threads it into the next call. This is the cheapest learner over the corpus, which is the point of the comparison.

The fold prompt as the algorithm. Equation 4.1 places all of the learning behavior in the prompt, so we state its contract explicitly. The system prompt specifies four constraints. First, the document is a distillation of transferable methodology and never a log of the claims seen; recording individual claims, verdicts, named entities, specific dates, or URLs is forbidden, and an illustrative case must be abstracted into a pattern. Second, the model edits in place so it must integrate a new observation into existing guidance, collapse two

observations that say the same thing, and subsume the n -th instance of a known pattern under the general rule instead of appending it. Third, a stable section skeleton is supplied so that successive folds edit the same structure rather than reinventing the layout each batch. Fourth, a set of scannability rules forbids run-on enumerations, requires cross-section de-duplication with each method stated once in its best home, and directs the model to cut filler a competent fact-checker already knows. A soft length target of W words is communicated in every fold prompt. The target is advisory and never enforced by the driver.

Consolidation passes. Incremental folding degrades readability even under the edit-in-place instruction, because guidance accretes across sections. We therefore schedule a separate consolidation call every c folds and once more at the end of the run. Consolidation sees the document alone with no batch attached. Its prompt asks for cross-document de-duplication, removal of self-evident guidance, and conversion of run-on enumerations into scannable checklists, while explicitly prohibiting new content, new examples, and any claim-specific fact. Every distinct technique, claim type, pitfall, and stopping rule must remain findable afterward, so consolidation removes repetition of a method and never its single canonical statement. Scheduling on a fixed cadence rather than triggering on a length budget keeps the schedule independent of how verbose any particular fold happened to be. Algorithm 1 shows the full learning process.

Output format and failure handling. The payload is a large free-text markdown document, so the model emits it between verbose sentinels rather than inside JSON, which avoids brittle escaping of the document body. Each response carries a changelog section and a strategy section and a response missing the strategy sentinels triggers one targeted repair call that restates the format requirement and quotes the previous response. If the repair also fails to parse, the fold returns `ok=false` and the *unchanged* input document is kept, so a malformed generation degrades to a no-op rather than corrupting the run. Failed folds are counted and reported. At the transport level, every model call is retried up to five times with exponential backoff, since a build of several hundred sequential calls will otherwise be aborted by a single transient provider error.

4.5.2. Inference

The inference-time consumer is a fact-checking agent whose entire configuration is the document, the claim, and two investigation tools. The baseline replaces the blueprint control layer, so the agent performs no blueprint selection, builds no verification graph, and maintains no blueprint session state. The planner has access to the same Web-Search (Section 4.3.3) and Media (Section 4.3.4) agents for task delegation and final verdicts are given by the same judge (Section 4.3.5).

Planning loop. Verification proceeds in at most R planning rounds. In each round the planner receives a system prompt containing its role, the two tool descriptions, the required response schema, and the full document delimited by explicit markers. The user prompt supplies the claim, the round index, an inventory of attached modalities with the exact reference tag of each image and video, the evidence gathered so far rendered as source and takeaways, and the history of previous tool calls with their outcomes. On the last round the prompt adds a directive to gather any remaining evidence now or to declare completion. The planner answers with strict JSON carrying its reasoning, a list of tool calls, and a done flag. Investigation stops when the planner sets done, when it requests no tool call, or when the round budget is exhausted. A failing sub-agent call must not lose the claim, so exceptions are caught per call and converted into an error plus a history entry that the planner can react to in a later round.

5. Experiments

Chapter 4 describes a pipeline that learns verification procedures from published fact-checks and routes each incoming claim to one of them. This chapter measures what that machinery is worth, following the second and third questions posed in Chapter 1. The first half asks whether conditioning verification on a per-claim strategy improves accuracy. The second half asks where the remaining error actually comes from, since a negative answer is only informative once the binding constraint is named.

The headline comparison holds every component of the pipeline fixed and varies only the guidance the planner receives. Three conditions run over the same 1000 claims. One routes each claim to a learned blueprint, one hands every claim the same distilled playbook, and one supplies no guidance at all. Two planner models execute each condition, and five judge models rescore each run, so a conclusion has to survive a change of model on either side before it is reported as one. Section 5.1 fixes the data, the models, and the settings shared by every run, and Section 5.2 reports the resulting grid together with a comparison against the published VERITAS baseline.

That grid returns a qualified negative. Learned blueprints help the weaker planner and hurt the stronger one, and the difference between those two effects is the only contrast in the table that is individually significant. The rest of the chapter treats that outcome as the starting point rather than the conclusion. Section 5.3 decomposes the error by removing one stage of the pipeline at a time, first routing, then retrieval, and finds that the verdict stage is close to saturated once the evidence is right. Section 5.4 closes with two components that were built because they looked promising and are reported because they did not pay off.

Three conventions govern how the numbers should be read. Mean squared error against the graded ground-truth label is the primary metric throughout, and accuracy accompanies it for interpretability alone. Nearly every comparison is paired over identical claims and read against a measured resampling floor rather than against zero. Comparisons that concern the judge alone are run by replaying recorded prompts, which removes retrieval variation from the contrast entirely. Section 5.1.6 and Section 5.1.7 state these in full.

5.1. Experiment Setup

5.1.1. Data

All experiments in this thesis run on VERITAS [110], a benchmark for multimodal claim integrity derived from published fact-check articles. Each sample pairs a short claim with its posting date (or fact-check date as fallback), zero or more attached media items (images / videos), the full text of the fact-check article that adjudicated it, and a link to the original review. Ground truth is a continuous integrity score in $[-1, 1]$, which grades the claim as a whole, so authentic footage under a misleading caption counts as  **Compromised**. VERITAS records finer sub-scores for textual veracity, context coverage, and per-item media authenticity and

Portion	Quarters	Claims	With media	Media items	Languages	Sources
Learning split	2025 Q1–Q4	4000	64.9 %	3070	51	123
Evaluation split	2026 Q1	1000	84.9 %	1013	41	88

Table 5.1.: VERITAS portions used in this work. “Sources” counts distinct fact-checking domains behind the review links.

contextualization but this thesis predicts and scores only the overall integrity value. Claims in the evaluation split are dated between January 1 and March 31, 2026, beyond the pretraining cutoff of the models employed, so verdicts must rest on retrieved evidence rather than parametric recall of the debunk.

The integrity score is mapped to discrete labels by thresholding. Scores of at least $1/3$ are  **Intact** and scores of at most $-1/3$ are  **Compromised** under the three-class scheme, with the interval between them left  **Unknown**. The seven-class scheme splits each direction into three confidence bands at multiples of $1/6$, which lets a system express how firmly it holds a verdict. Results are reported as mean squared error against the ground truth score under the seven-class mapping, with the three-class coarsening alongside for interpretability. Ground truth values are coarsely quantized in practice and cluster on ± 0.92 , ± 1.0 , and ± 0.83 , so squared error is driven mostly by direction errors and only secondarily by calibration.

VERITAS is published in quarterly splits of 1000 claims each, balanced by construction into roughly half  **Intact** and half  **Compromised** claims plus a few  **Unknown** cases. Table 5.1 summarizes what this thesis uses. Blueprint learning draws on the four 2025 quarters and the 2026 Q1 split is held out as the evaluation set throughout, so no claim seen during learning reappears at test time.

The evaluation split is markedly more multimodal than the material it was learned from. 849 of its 1000 claims carry media, against 64.9 % in the learning split, and its 1013 media items divide into 525 images and 488 videos. Media rarely arrives clean, since 37.4 % of the items are labeled inauthentic and 47.9 % miscontextualized. The benchmark’s own record of which aspect decided each verdict tells the same story. Media contextualization decides 453 claims and claim-level context coverage a further 354, while textual veracity alone decides only 193. Media reasoning therefore dominates the benchmark, which motivates the components described in Chapter 4. Coverage is broad in language and provenance as well, spanning 41 languages and 88 fact-checking domains, led by English, Spanish, and Hindi and by the regional AFP desks, Lead Stories, StopFake, and Newschecker.

Three properties of the construction shape how the results should be read. First, the class balance is engineered through rectification, since 492 of the 499 intact claims are rewritten versions of a debunked original. Both classes share topic, phrasing, and media, which makes the benchmark harder than its balance suggests. Second, manipulated media does not by itself imply a compromised claim, because a claim that correctly identifies an image as fabricated is truthful about it. 156 intact claims in the evaluation split carry an inauthentic media item, so a system that reasons from detected manipulation straight to a verdict will fail all of them. Third, the fact-check article names the answer and is withheld from the pipeline at inference time. It is used only in the offline article analysis that supervises blueprint learning and in the oracle-evidence ablations of Section 5.3.2.

5.1.2. Models

Nine models appear in this chapter, and they occupy three roles that never mix. Table 5.2 lists each one with the role it fills. A run is characterized by a single *pipeline model*, which drives the planner, the web-search

Model	Access	Knowledge Cutoff	Role
GEMINI 3 FLASH [39]	API	January 2025	Pipeline model, judge, geo-location tool
GEMMA 4 31B [128]	self-hosted	January 2025	Pipeline model, judge
CLAUDE OPUS 4.6 [6]	API	May 2025	Judge
QWEN 3.5 35B [106]	self-hosted	Not disclosed [†]	Judge
QWEN 3.5 397B [106]	self-hosted	Not disclosed [†]	Judge
CLAUDE OPUS 5 [8]	API	May 2026	Judge, ceiling experiments only
QWEN 3.6 35B [107]	self-hosted	Not disclosed [‡]	Judge, ceiling experiments only
CLAUDE OPUS 4.8 [7]	API	January 2026	Offline: blueprint synthesis and merging, Strategy.md
GEMINI 3.5 FLASH [40]	API	January 2025	Offline: article analysis, strategy fingerprints

Table 5.2.: Models used in this work, grouped by role. Pipeline models drive every agent inside one run; judges rescore finished runs over recorded inputs; offline models run once during blueprint and strategy construction and never at inference time. [†] Alibaba publishes no pretraining cutoff for any Qwen 3.5 checkpoint. Evaluation material in the model cards postdates November 2025 (35B) and February 2026 (397B), which bounds the release rather than the training corpus. [‡] Released in April 2026.

agent, the media agent, the per-source summarizer, and the blueprint selector alike. Holding those stages to one model keeps the comparison between conditions about the guidance the planner receives rather than about which internal component happened to be upgraded. Two pipeline models are used. GEMINI 3 FLASH is a hosted model at the fast end of the current frontier, and GEMMA 4 31B is an open-weight model served locally on the group’s own hardware. They span roughly an order of magnitude in capability, which is what makes the blueprint effect testable against planner strength rather than measurable at one point only.

The *judge* is treated separately because it is the only stage whose output enters the metric. Every finished run is rescored by five judges over the recorded judge inputs, so the evidence stays byte-identical and the judge is the only thing that varies. The five are GEMINI 3 FLASH, CLAUDE OPUS 4.6, GEMMA 4 31B, QWEN 3.5 35B, and QWEN 3.5 397B, which cover two hosted APIs and three self-hosted checkpoints across three parameter scales. A verdict-stage result that reverses between them is a property of one judge and is reported as such. CLAUDE OPUS 5 and QWEN 3.6 35B extend that panel in the ceiling experiments of Section 5.3.2 and Section 5.4.2, where the question is how much headroom a stronger verdict stage still has and the wider spread is worth the extra cost.

The third role is *offline construction*, which runs once before any evaluation and never at inference time. CLAUDE OPUS 4.8 synthesizes blueprints from clusters, merges them during consolidation, and folds the corpus into the Strategy.md document of the single-strategy baseline. Synthesis is a few hundred calls over long inputs whose output is then reused across a thousand claims, so quality there is paid for once and a frontier model is affordable. Article analysis and fingerprint extraction, which turn each training article into the structured record those synthesizers consume, run on GEMINI 3.5 FLASH. Fingerprints are embedded with OpenAI text-embedding-3-large at $d = 3072$, and the same embeddings train the linear selection probe.

Knowledge cutoffs matter here because the evaluation claims are dated between January and March 2026. A pipeline model whose training data reaches into that window could recall a published debunk instead of retrieving evidence for it, which would inflate the results without improving the system. Neither pipeline model is exposed, since both are cut off in January 2025, a year before the evaluation quarter. Among the offline models, CLAUDE OPUS 4.8 reaches January 2026 but only ever reads 2025 articles, so it cannot transfer a 2026 verdict it never saw. One genuine exposure remains. CLAUDE OPUS 5 is cut off in May 2026 and therefore postdates every evaluation claim. It is confined to the two ceiling experiments and never plans,

```
claim_type: media_authenticity
decisive_actions: reverse_image_search
evidence_types: reverse_image_search, source_lookup, web_search, expert_consultation
action_sequence: reverse_image_search, web_search, expert_consultation
steps: Performed a reverse image search on keyframes of the viral video.; Identified and analyzed a May 2015 ABP News
      report containing the video.; Conducted searches to find additional media reports from May 2015 confirming the Indore
      police anti-crime drive.; Contacted the Indore police for official confirmation and further details.
process_richness: full
modality_flags: has_video, is_multimodal
```

Figure 5.1.: Strategy fingerprint for the claim *This video shows Indore police beating arrested people during an anti-crime drive in May 2015*. The structured fields describe the investigative process alone, while the free-text steps retain some topical detail which are reduced to a minimum. Each step is truncated to 120 characters and at most five are kept. Line breaks in the steps field were added for readability.

retrieves, or scores a main-results run. Its gold-evidence numbers land within 0.003 MSE of GEMINI 3 FLASH, which bounds any memorization advantage far below the effects it is used to measure. The exposure should nonetheless be kept in mind wherever an CLAUDE OPUS 5 column appears.

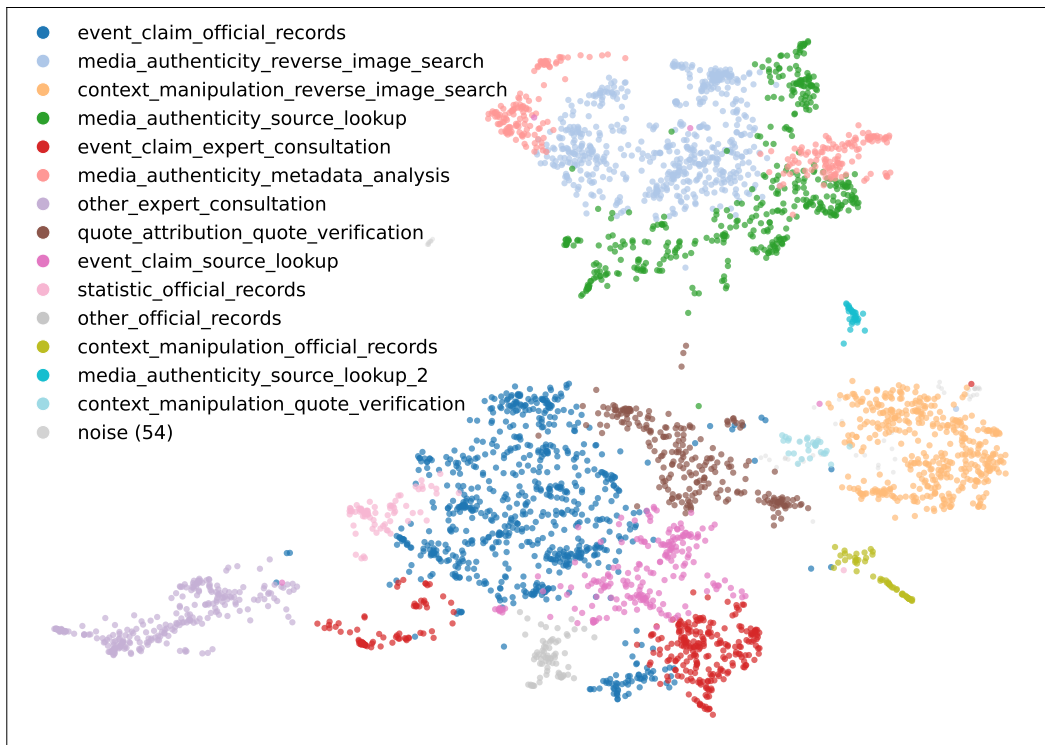
5.1.3. The Learned Blueprint Pool

Blueprints are generated out of the VERITAS 2025 data. First, the articles are analyzed and strategy fingerprints are extracted. Figure 5.1 shows an example of an extracted fingerprint. Then, for clustering, the dimensionality of embedded strategy fingerprints is reduced to 50 using UMAP. HDBSCAN is configured to use a minimal cluster size of 20 (i.e. at least 20 claims per cluster) and the excess-of-mass cluster selection method, which selects the clusters that are most stable and persistent across the hierarchy tree. The resulting clusters are then synthesized into blueprints using CLAUDE OPUS 4.8 with up to 30 representatives per cluster.

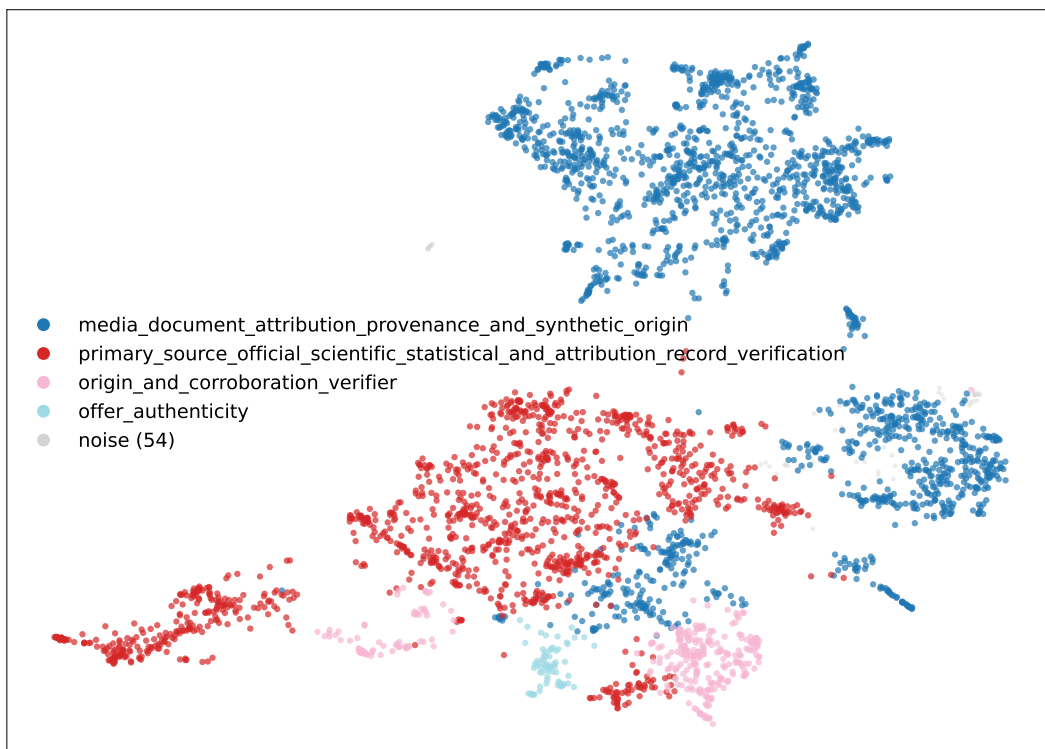
Of the 4,000 claims, 291 are excluded from the blueprint generation because their fact-check articles report only a verdict without any investigative steps, leaving nothing to fingerprint. Clustering the remaining 3,709 claims produces 14 clusters covering 98.5% of them and leaves only 1.5% labeled as noise. No cluster is incoherent enough to be split, and the largest one lands exactly on the 20% size cap with 731 claims. Consolidation then merges the eight media-oriented clusters into a single provenance blueprint covering 1,983 claims, and four record-oriented clusters into one covering 1,299, while the remaining two clusters survive unmerged. Four blueprints therefore carry traffic, two of which cover 90% of all clustered claims, and a fifth generic blueprint absorbs routing failures. Figure 5.2 shows the embedded space before and after this compression. Finally, the linear selection probe is fitted and validated with stratified cross-validation. Its confidence threshold is set to 0.8, since the probe is accurate in roughly 94% of the cases that clear that bar and drops toward chance below 0.6. Whether the selector routes claims to these blueprints reliably, and whether the routing decision affects the verdict at all, is measured in Section 5.3.1. Appendix A.7 reproduces all five blueprints in full, exactly as the pipeline loaded them.

5.1.4. Compared Systems

Three conditions are compared, and they differ in exactly one respect, namely what procedural knowledge the planner is given and how that knowledge is structured. Everything downstream of the planner is shared. All three delegate to the same web-search and media agents under the same retrieval budget, and all three hand their collected evidence to the same judge under the same three-sample protocol. Each condition is



(a) Visualization of strategy space after clustering, reduced with UMAP. Cluster names join the most frequent claim type with the action that most often proved decisive.



(b) The same points and the same projection, colored by the blueprint each claim was assigned after synthesis. Labels are the blueprint names.

Figure 5.2.: Claim space before and after blueprint synthesis. Color change alone shows what synthesis merged. Fourteen clusters reduce to four blueprints, two of which cover 90% of all clustered claims.

executed end to end with each of the two pipeline models, which yields six full runs over the 1000 evaluation claims, and every run is then rescored by the five judges of Section 5.1.2.

Blueprints. The full system of Chapter 4. Each claim is routed to one of the five learned blueprints described in Section 5.1.3 by the hybrid selector, which takes the linear probe’s answer when its confidence reaches 0.8 and defers to the LLM selector otherwise. The selected blueprint supplies the checks to resolve, the actions available for resolving them, and the branching structure among them, and the planner traverses that graph until it finalizes or runs out of nodes. The iteration budget is a property of the blueprint and ranges from four for the generic fallback to nine for the media provenance blueprint.

Single strategy. The baseline of Section 4.5. The blueprint pool and the entire control layer around it are replaced by one free-text document, `Strategy.md`, which every claim receives unchanged. No selection happens, no graph is built, and no blueprint session state is kept. The document is distilled by CLAUDE OPUS 4.8 from the same 3,709 article analyses the blueprints were clustered from, so both conditions see identical training material and differ only in how it is represented. Construction runs one epoch over batches of ten records with an advisory target of 2,500 words and a consolidation pass every five folds, which is 371 folds and 75 consolidations with no parse failures. The resulting document is roughly 4,300 words in eight sections. Its planner runs a flat loop of at most two rounds, delegating in parallel within each. Appendix A.8 displays `Strategy.md` in full, exactly as the pipeline loaded it.

Empty strategy. The same standalone agent, the same two-round loop, and the same tools as the single-strategy condition, with the document replaced by a two-sentence placeholder that supplies no methodology.

```
# Strategy.md - (empty baseline)
```

```
No fact-checking playbook is provided. Rely on general best practices to verify the  
claim with the available tools, then reach a verdict from the evidence.
```

This arm isolates the value of the learned content from the value of the scaffolding that delivers it. Whatever the agent achieves here comes from the model’s own priors plus the tools, so the gap between it and the single-strategy condition prices the distilled document, and the gap between the single-strategy and blueprint conditions prices the routed structure on top of it.

One asymmetry between the arms is worth naming, because it does not favor the conclusion drawn later. The blueprint condition is given a larger execution budget than either baseline, since its graph permits four to nine iterations against the two planning rounds the strategy agent receives, even though it spends only 3.23 iterations on average. Per delegated retrieval task the budget is identical across all three, so the extra allowance buys more delegations rather than deeper search on any one of them. The blueprint arm is therefore the better-resourced condition wherever it loses.

A fourth reference point sits outside this grid. Section 5.2.1 compares against the pipeline published with VERITAS, re-executed on the same claims rather than quoted from the paper. That comparison is not like-for-like and is read accordingly, as the caption of Table 5.5 sets out.

Component	Setting	Value
All roles	temperature	1.0
All roles except summarization	reasoning effort	provider default
Planner, media, judge	response budget	64k tokens
Blueprint selector	response budget	8k tokens
Source summarization	response budget, thinking	5k tokens, disabled
Retrieval	search rounds per task	2
Retrieval	queries per round	3
Retrieval	results kept per query	3
Judge	independent samples	3, averaged
Blueprint selection	method	hybrid
Blueprint selection	probe confidence threshold	0.8

Table 5.3.: Configuration held fixed across all reported runs. No run sets an explicit reasoning effort, so every backend uses its own default: adaptive thinking at high effort for Claude, a dynamic thinking budget for Gemini, and the chat-template default for the self-hosted models. Response budgets are joint ceilings on Anthropic, where reasoning and answer tokens draw from the same allowance. Deviations are named at the experiment that introduces them.

5.1.5. Pipeline configuration

Table 5.3 lists every setting held fixed across the reported runs. All roles sample at temperature 1.0, which follows the recommendation for the reasoning models used and keeps the judge’s repeated samples independent. Response budgets differ by role because reasoning tokens count against the same limit, and per-source summarization is the one call that runs without thinking, since it only reduces a document to a set of bullet points. The retrieval budget is capped for reasons of cost and wall-clock time. Anything varied by an experiment is stated where that experiment is introduced, and everything else in the table stays as shown.

5.1.6. Metrics

The primary metric is the mean squared error between the predicted integrity score and the ground truth score, both on $[-1, 1]$. Squared error suits a continuous target and penalizes a confident verdict in the wrong direction far more heavily than an unconfident one. Mean absolute error is reported alongside it. Predictions enter the metric before discretization. The judge’s three samples are averaged as numeric labels, and that raw average is scored rather than the seven-class label nearest to it. Scoring the snapped label would charge a prediction with error introduced by the label grid rather than by the verdict, which is a property of the reporting scheme. Single-sample runs fall back to the label’s own value, for which the two quantities coincide.

Classification metrics accompany the regression numbers for interpretability. Accuracy and the confusion matrix are reported over the seven-class scheme and again over a three-class coarsening. One detail of that coarsening affects how the tables read. Both *rather uncertain* labels collapse to **Unknown** rather than to their nominal direction, so a system receives credit for a directional call only when it committed to one.

5.1.7. Evaluation protocol

Nearly every claim in this chapter is a paired one. Two configurations are run over an identical set of claims, and the reported quantity is their difference on that set rather than two independently sampled numbers placed side by side. Pairing removes claim difficulty from the comparison, which matters here because the per-claim squared errors are heavily skewed and a few confident direction errors move an unpaired mean more than a real effect does. Each comparison also reports how many claims improved, how many worsened, and how many changed direction, since a small net difference can hide substantial churn underneath.

Isolating one change is difficult when the pipeline is both expensive and stochastic, so comparisons that concern the judge alone are run by replay. Every finished run stores each claim’s judge input in its trace, and the judge is re-invoked over that stored input with exactly one setting altered. Replaying the recorded prompt is byte-identical to what was originally judged, so any delta is attributable to the judge and never to a difference in what retrieval happened to find that day. Comparisons that touch retrieval or planning cannot be replayed and are run end to end.

Sampling at temperature 1.0 means no run reproduces exactly, so differences are read against a measured resampling floor (the spread of an unchanged run re-judged) rather than against zero. That floor is measured directly, by re-judging a run with nothing changed and taking the spread across repetitions. Runs are resumable and write one JSON line per claim, so an interrupted run continues without repeating completed work.

5.2. Main Results

Table 5.4 reports the full grid with two planner models, three strategy conditions, and five judge models scoring every run. The effect of learned blueprints reverses with planner capability. Averaged over the five judges, blueprints improve MSE for the GEMMA 4 31B planner from 0.401 to 0.347 and degrade it for the GEMINI 3 FLASH planner from 0.318 to 0.358. Both movements are small next to the spread that the choice of judge introduces, so neither number carries the claim on its own. Consistency does. All five judges rank blueprints ahead of the empty strategy under GEMMA 4 31B and behind it under GEMINI 3 FLASH, with no dissent in either direction. The single-file strategy then separates the two ingredients of the blueprint condition. On GEMMA 4 31B it already recovers 0.042 of the 0.054 gain, which places most of the benefit in the distilled guidance rather than in the routing of it. On GEMINI 3 FLASH it is nearly neutral at +0.006 against the +0.040 that blueprints cost, so the damage is specific to the routed structure. Figures 5.3 and 5.4 show confusion matrices for the runs whose planner and judge are the same model. Appendix A.6 provides the remaining judges and the three-bin discretization.

Retrieval quality varies more across claims than any method effect reported here, so the reversal could in principle reflect one arm retrieving better than the other. It does not. The two planners leave unfetchable stubs at indistinguishable rates, 23.4% against 23.1% under GEMMA 4 31B and 17.9% for both GEMINI 3 FLASH arms. A claim’s failure count also predicts its squared error no better than the sheer volume of evidence retrieved does, at +0.12 against +0.11. Restricting to the 468 claims where retrieval recovered most of the reachable ground-truth findings leaves the single-strategy interaction at -0.084 , against -0.091 on the full set. Cleaner retrieval therefore does not shrink the effect, though halving the sample makes the restricted figure a weak check on its own.

Read in both directions, the reversal suggests that a blueprint substitutes for a planner’s own procedure rather than adding to it. A model with a weaker internal procedure gains from being told what to do, and a model

Planner Model	Method	Judge Model	MSE ($\downarrow$)	MAE ($\downarrow$)	Acc. 7 ($\uparrow$)	Acc. 3 ($\uparrow$)
GEMINI 3 FLASH	Blueprints	GEMINI 3 FLASH	0.307	0.370	30.1%	75.9%
		CLAUDE OPUS 4.6	0.306	0.379	35.5%	70.7%
		GEMMA 4 31B	0.421	0.393	49.2%	70.3%
		QWEN 3.5 35B	0.393	0.441	28.0%	65.7%
		QWEN 3.5 397B	0.362	0.424	31.7%	64.7%
	Single Strategy	GEMINI 3 FLASH	0.285	0.335	38.9%	78.3%
		CLAUDE OPUS 4.6	0.247	0.341	36.9%	73.5%
		GEMMA 4 31B	0.380	0.368	48.9%	72.8%
		QWEN 3.5 35B	0.366	0.425	29.0%	66.6%
		QWEN 3.5 397B	0.344	0.404	33.7%	66.2%
	Empty Strategy	GEMINI 3 FLASH	0.276	0.331	38.6%	78.2%
		CLAUDE OPUS 4.6	0.249	0.342	37.1%	73.8%
		GEMMA 4 31B	0.374	0.365	48.7%	72.7%
		QWEN 3.5 35B	0.350	0.416	31.3%	66.8%
		QWEN 3.5 397B	0.342	0.409	32.1%	66.6%
GEMMA 4 31B	Blueprints	GEMINI 3 FLASH	0.303	0.370	32.1%	74.6%
		CLAUDE OPUS 4.6	0.271	0.363	35.1%	70.4%
		GEMMA 4 31B	0.412	0.403	46.2%	68.3%
		QWEN 3.5 35B	0.386	0.441	28.3%	65.1%
		QWEN 3.5 397B	0.361	0.423	32.1%	64.6%
	Single Strategy	GEMINI 3 FLASH	0.347	0.394	33.9%	71.8%
		CLAUDE OPUS 4.6	0.298	0.376	36.7%	69.7%
		GEMMA 4 31B	0.389	0.386	48.5%	71.1%
		QWEN 3.5 35B	0.391	0.442	29.6%	65.6%
		QWEN 3.5 397B	0.372	0.421	33.7%	66.6%
	Empty Strategy	GEMINI 3 FLASH	0.365	0.409	30.8%	71.4%
		CLAUDE OPUS 4.6	0.317	0.398	34.9%	66.9%
		GEMMA 4 31B	0.455	0.430	50.2%	67.4%
		QWEN 3.5 35B	0.450	0.486	25.9%	61.2%
		QWEN 3.5 397B	0.418	0.463	30.5%	62.6%

Table 5.4.: Full VERITAS 2026 Q1 results across two planner models, three strategy conditions, and five judge models. Blueprints improve over the empty strategy for the GEMMA 4 31B planner and degrade it for GEMINI 3 FLASH.

with a stronger one loses by being constrained to someone else’s method. Once the procedure is supplied externally, which model plans barely matters, and the two planners end up 0.011 MSE apart. That is no argument for the cheap planner, though. GEMMA 4 31B with blueprints (0.347) still trails GEMINI 3 FLASH with no strategy at all (0.318), so blueprints converge the planners at a level below what the better one reaches unaided.

Both the gain and the cost are confined to direction errors, measured here under the judge matching the planner model. On the GEMMA 4 31B planner, blueprints cut the direction-flip rate by 1.31 percentage points

Method	Judge Model	MSE ($\downarrow$)	MAE ($\downarrow$)	Acc. 7 ($\uparrow$)	Acc. 3 ($\uparrow$)
VERITAS [110] Baseline [†]	GEMINI 3 FLASH	0.382	0.286	65.9%	87.8%
	CLAUDE OPUS 4.6	0.281	0.303	51.9%	82.6%
	GEMMA 4 31B	0.491	0.373	55.4%	82.9%
	QWEN 3.5 35B	0.551	0.441	44.6%	78.8%
	QWEN 3.5 397B	0.429	0.359	53.2%	82.9%
Ours (Blueprints)	GEMINI 3 FLASH	0.307	0.370	30.1%	75.9%
	CLAUDE OPUS 4.6	0.306	0.379	35.5%	70.7%
	GEMMA 4 31B	0.421	0.393	49.2%	70.3%
	QWEN 3.5 35B	0.393	0.441	28.0%	65.7%
	QWEN 3.5 397B	0.362	0.424	31.7%	64.7%
Ours (Empty Strategy)	GEMINI 3 FLASH	0.276	0.331	38.6%	78.2%
	CLAUDE OPUS 4.6	0.249	0.342	37.1%	73.8%
	GEMMA 4 31B	0.374	0.365	48.7%	72.7%
	QWEN 3.5 35B	0.350	0.416	31.3%	66.8%
	QWEN 3.5 397B	0.342	0.409	32.1%	66.6%

Table 5.5.: Comparison to the VERITAS baseline on 2026 Q1, same claims and same targets. All our rows use the GEMINI 3 FLASH planner. The comparison is not like-for-like because the baseline uses a single web-search tool with a smaller query budget and a single judge pass, while our pipeline adds media-forensic tools, a larger retrieval budget, and three-sample judge aggregation. Our system wins MSE against every judge and loses Acc. 7 and Acc. 3 against every judge, a split that Section 5.2.1 attributes to confidence policy. [†] Re-executed, not taken from the paper.

at roughly 3.0 squared error per flip, which accounts for 0.040 of that run’s 0.043 MSE gain. Accuracy within the correct direction is unchanged, and exact-bucket accuracy does not improve. Under GEMINI 3 FLASH the pattern inverts. Blueprints add 1.32 percentage points of flips, and 8 of the 13 added flips are hedges into the neutral band. That hedging has one identifiable source, since a blueprint finalizing with a required check still unresolved lands in the neutral band far more often than one closing every check. The rate is 13.4% against 2.4% under GEMINI 3 FLASH, and 32.6% against 10.6% under GEMMA 4 31B. Both planners show the gap, so the driver is the open check rather than the model reading it. Claims where only the blueprint hedges cost +0.025 MSE under GEMMA 4 31B and +0.014 under GEMINI 3 FLASH, so the hedging tax falls on both and the reversal comes from the benefit side.

Variation across judges exceeds any method effect. The spread within a single cell reaches 0.14 MSE, roughly two and a half times the largest method effect. GEMMA 4 31B as judge ranks worst on MSE in five of the six planner–method cells and best on Acc. 7 in all six. It commits to a bucket more readily and pays for it when wrong, while GEMINI 3 FLASH and CLAUDE OPUS 4.6 hedge toward the middle and win MSE, MAE, and Acc. 3. Reporting a single judge would therefore have supported either conclusion about judge quality, depending on the metric chosen.

One further asymmetry runs against the blueprint condition. It is the most expensive arm under both planners, issuing the most model calls under GEMMA 4 31B and consuming the most input tokens under GEMINI 3 FLASH, where a run costs \$175 against \$130 for the empty strategy and takes 4.4 minutes per claim against 3.4. Appendix A.2 reports the measured figures. The extra spend buys an improvement only on the weaker planner, and on the stronger one it purchases the degradation reported above.

Ground Truth \ Prediction	-1	-2/3	-1/3	0	+1/3	+2/3	+1
-1	353	9					13
-2/3	88	16			2		14
-1/3	3	1					
0	1						
+1/3	2	1					8
+2/3	24	3			8		117
+1	43	6		1	5		282

(a) GEMINI 3 FLASH Baseline

Ground Truth \ Prediction	-1	-2/3	-1/3	0	+1/3	+2/3	+1
-1	252	67			1	15	40
-2/3	58	35				12	15
-1/3	2					2	
0	1						
+1/3						2	9
+2/3	7	9	1	3	1	18	113
+1	19	28		3	1	37	249

(b) GEMMA 4 31B Baseline

Ground Truth \ Prediction	-1	-2/3	-1/3	0	+1/3	+2/3	+1
-1	143	186	10	13	11	11	1
-2/3	44	35	7	13	4	13	3
-1/3	2	1	1				
0	1						
+1/3		1		1		7	2
+2/3	2	3	6	12	22	79	28
+1	8	16	8	16	48	198	43

(c) GEMINI 3 FLASH Blueprints

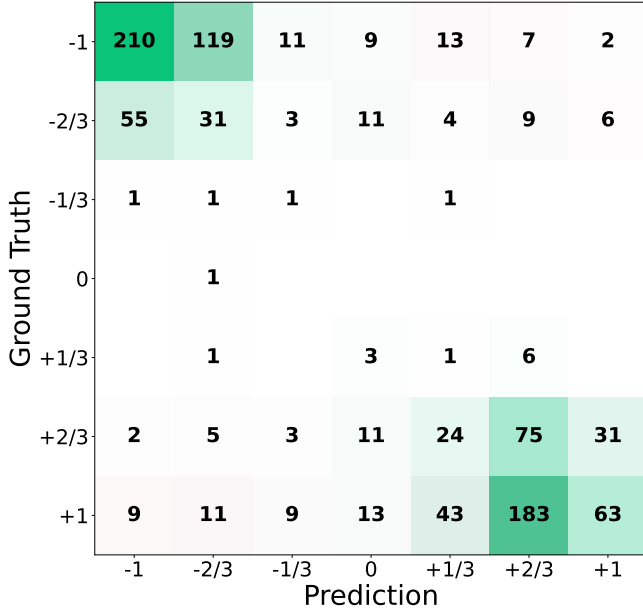
Ground Truth \ Prediction	-1	-2/3	-1/3	0	+1/3	+2/3	+1
-1	290	13	1	57	3	9	2
-2/3	76	10		21		9	4
-1/3	2	2					
0	1						
+1/3				1	1	2	7
+2/3	9	2	1	36	4	26	73
+1	28	4		90	23	57	135

(d) GEMMA 4 31B Blueprints

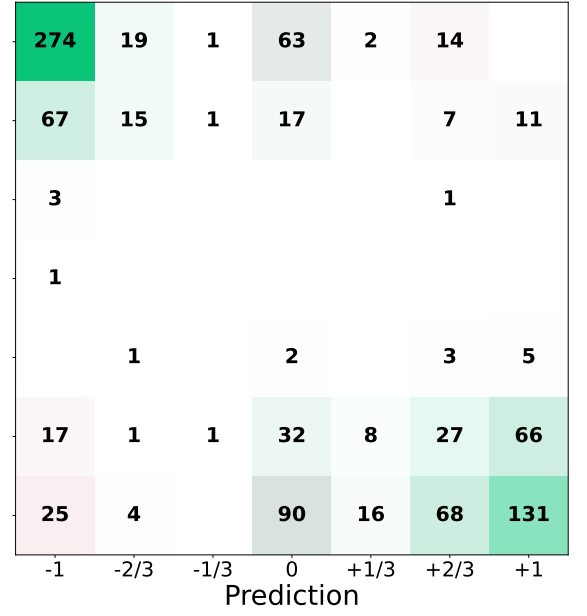
Figure 5.3.: Confusion matrices for the VERITAS baselines in (a) and (b). (c) and (d) show confusion matrices for the blueprint runs where planner and judge model are the same. Figure A.8 shows the same setting with 3-bin discretization.

5.2.1. Comparison to the VeriTAS baseline

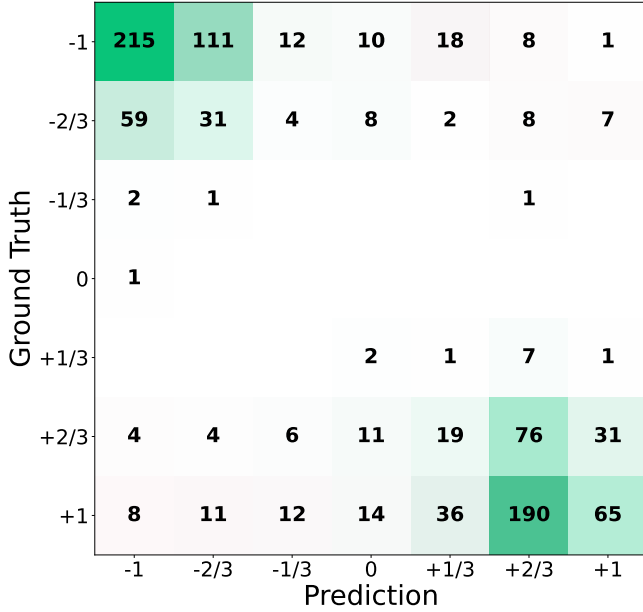
Table 5.5 places our pipeline next to the VERITAS baseline [110] on the same 1000 claims with the same ground-truth targets. Our empty-strategy configuration reduces MSE against all five judges, and the blueprint



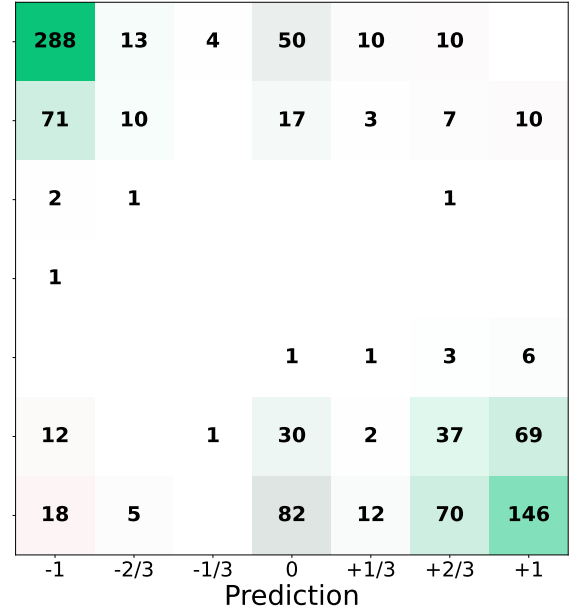
(a) GEMINI 3 FLASH Empty Strategy



(b) GEMMA 4 31B Empty Strategy



(c) GEMINI 3 FLASH Single Strategy



(d) GEMMA 4 31B Single Strategy

Figure 5.4.: Confusion matrices for the empty-strategy runs where planner and judge model are the same in (a) and (b). (c) and (d) contain the confusion matrices for the single-strategy runs in the same setting. Figure A.8 shows the same setting with 3-bin discretization.

configuration does so against four, losing only to the CLAUDE OPUS 4.6 baseline. Neither configuration wins Acc. 7 or Acc. 3 against any judge. MAE is split with our empty-strategy rows improving on the baseline against GEMMA 4 31B and QWEN 3.5 35B and trailing it against the other three. Against the GEMINI 3 FLASH judge the blueprint MSE difference is -0.075 paired, while Acc. 7 and Acc. 3 are 35.8% and 11.9% lower

Claim 103621: O vídeo mostra uma manifestação em Madri em solidariedade ao Irã e contra os ataques de Israel e dos Estados Unidos.

EN (The video shows a demonstration in Madrid in solidarity with Iran and against the attacks by Israel and the United States.)

ground truth **compromised** (certain)

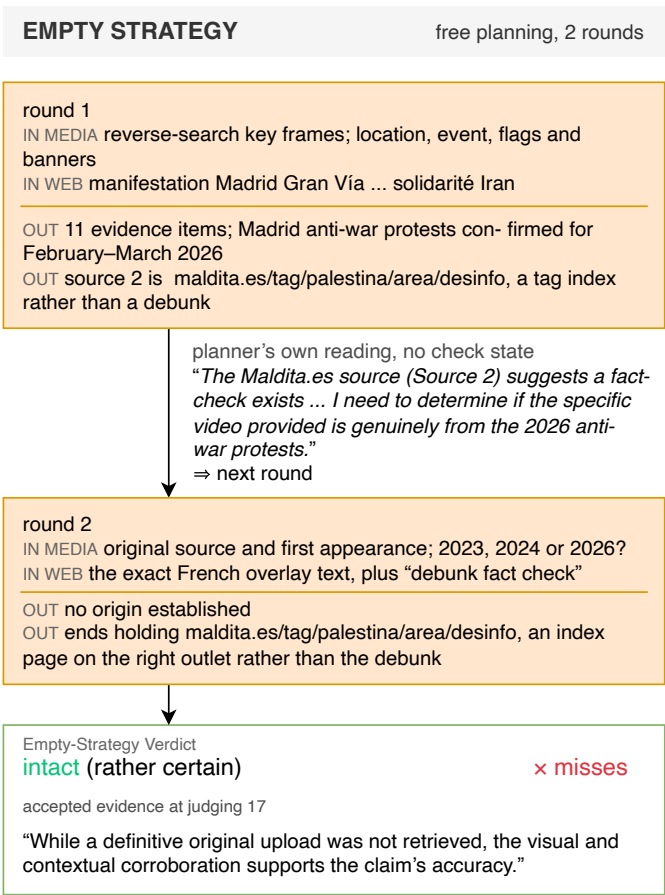
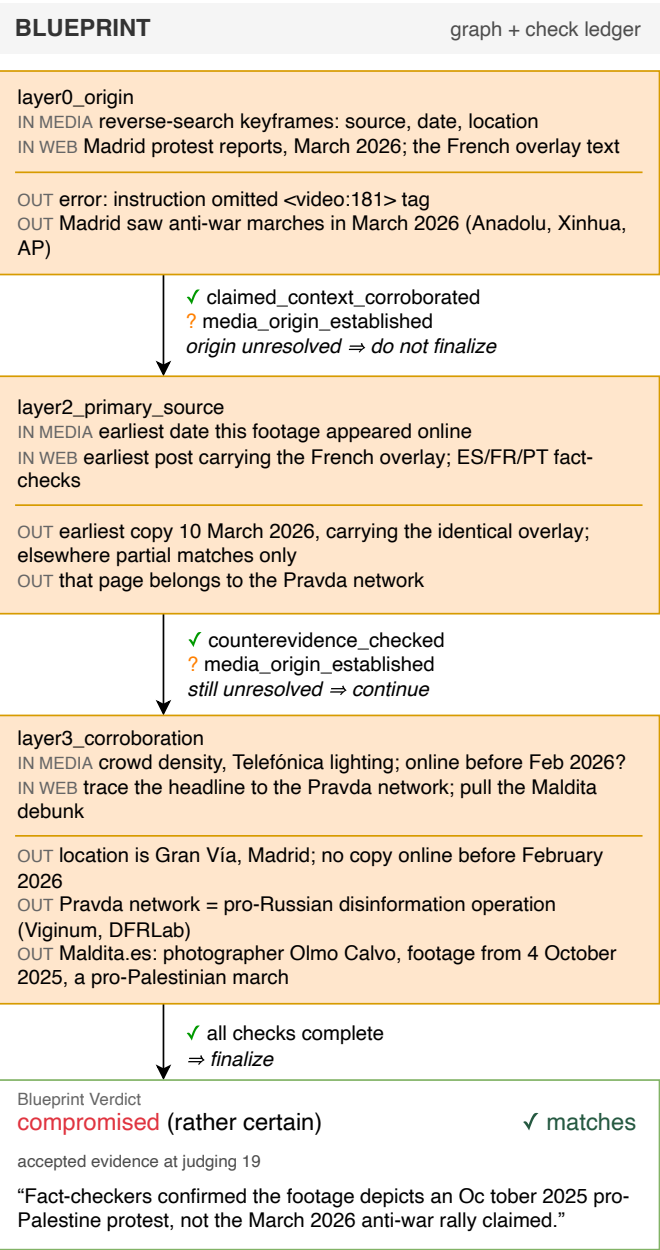


Figure 5.5.: **An open check buys two further retrieval passes.** Both arms confirm that mass anti-war protests took place in Madrid in March 2026, and the empty-strategy arm accepts that as verification of the attached footage. Three clauses stop the blueprint doing the same. `media_origin_established` counts “only an EXACT same-frame/same-image match ... as evidence about THIS media”; `layer1_synthesis` finalizes only “if origin and claimed context are already established”; and `layer3_corroboration` names the failure in advance, “report the provenance as unverified rather than assuming a false context”. Note that pass one’s media task was wasted by a missing `<video:181>` tag, and the ledger is what catches that and retries.

respectively. Two asymmetries qualify the comparison before any interpretation. The baseline is restricted to a single web-search tool with a smaller query budget, while our pipeline adds media tools and a larger retrieval budget, which favors us. The baseline is also single-pass, while our judge aggregates three samples, which favors us on MSE; re-scoring our predictions in the baseline’s discrete output space changes MSE by at most 0.010, so this accounts for under 10% of the gap.

Direction reliability runs the other way. Across our six runs the flip rate spans 7.1% for GEMMA 4 31B with blueprints to 9.0% for GEMINI 3 FLASH with a single strategy, while the baselines span 10.3% for CLAUDE OPUS 4.6 to 18.8% for QWEN 3.5 35B. The two ranges do not overlap, so every one of our configurations is more reliable on direction than every baseline. This margin comes from three-sample judge aggregation rather than from richer retrieval, since re-judging the same evidence with a single judge sample raises our flip rate to 11.0%, in line with the baselines. The Acc. 7 and Acc. 3 deficits reflect how confidently each system reports. Our runs commit to a *certain* label on 27.7% to 62.8% of claims, where the baselines commit on 49.4% to 94.8% and use the *unknown* bin on at most 0.6%. Because the benchmark coarsens both *rather uncertain* labels into *unknown*, each hedged step forfeits a 3-class hit even where the direction is correct. That this is a reporting policy rather than weaker evidence is confirmed by AUC, which is invariant to confidence with our GEMINI 3 FLASH configurations reaching 0.928 to 0.940 against the matched baseline’s 0.898.

For external context, the strongest published DEFAME [16] configuration on this split reaches 0.282 MSE and 88.3% three-class accuracy with CLAUDE OPUS 4.6¹. That figure is quoted rather than re-executed, and it is scored in the benchmark’s discrete output space against our three-sample average, so it is not directly comparable to the rows above. The pattern it shows is the familiar one. Our empty-strategy configuration beats it on MSE, 0.249 against 0.282, and trails it on three-class accuracy, 73.8% against 88.3%, which is the trade the confidence policy produces against the VERITAS baseline as well.

5.2.2. Qualitative Analysis

Aggregate scores cannot show what the blueprint does to a run. This section traces three claims where the two arms retrieve overlapping evidence and return opposite verdicts. Two go the blueprint’s way and one does not. The same mechanism is responsible for all three. Each case reproduces under a second planner and survives re-judging, so the divergence reflects the evidence each arm assembled rather than a judge sample. In both winning cases the empty-strategy arm establishes a *background* fact, that a Madrid protest occurred, that a Rio parade honored the president, then treats it as verification of the *specific* media the claim attaches. The blueprint cannot make that substitution since its required checks are separate cells, and a background fact closes only one of them. The unresolved cell is what buys the retrieval pass that reaches the decisive source (Figure 5.5).

Figure 5.6 is the counter-example, and it passes the three tests in reverse. The claim presents September 2025 footage as a current January 2026 event and misnames the agents as ICE. Both active checks are satisfied by establishing that the footage really is of that Chicago chase; neither asks whether the agency is named correctly, and recency is never posed at all. Both come back supported and the finalize exit fires after one pass. That this is structural rather than a retrieval miss is clearest in the ledger entry itself, which dates the footage to “late September 2025” while scoring the check supported, on a claim posted 13 January 2026. The empty arm’s planner reached the same conclusion, but its verdict was rescued by a judge holding four articles headlined *Border Patrol agents chase cyclist*, which its query phrasing had surfaced.

¹<https://veritas.mai.informatik.tu-darmstadt.de/leaderboard>

Claim 103074: Görüntüler, Chicago’da kendilerine laf atan bir kişiyi kovalayan ancak yakala- mayı başaramayan ICE göçmen polislerini göstermektedir.

EN The footage shows ICE immigration police in Chicago chasing a person who heckled them but failing to catch him.

posted 13 January 2026

ground truth **compromised** (certain)

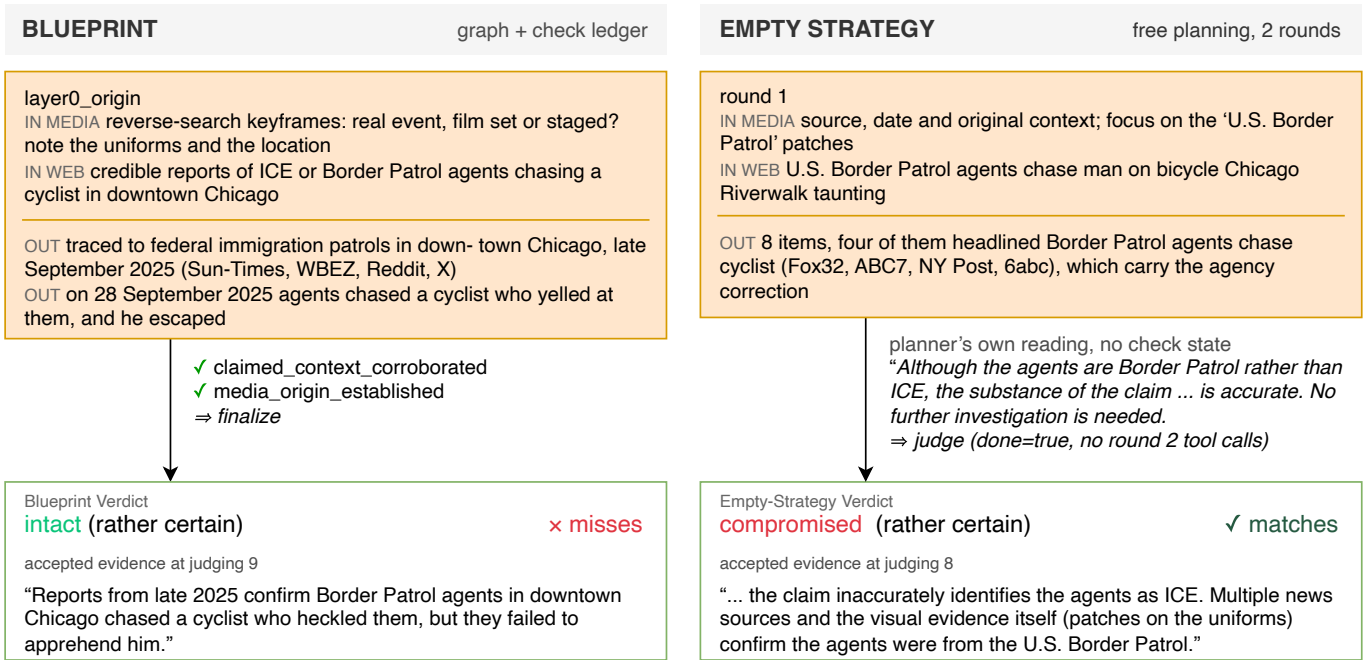
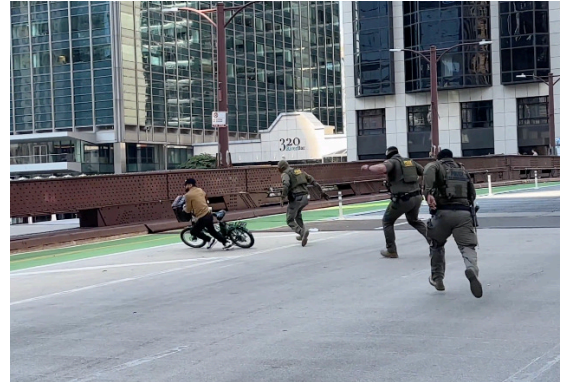


Figure 5.6.: **Checks that can be satisfied without touching what is wrong with the claim.** September 2025 footage is circulated as a current January 2026 event, and the agents are U.S. Border Patrol rather than ICE. Both of the blueprint’s active checks ask only whether the footage really is of that Chicago chase, both come back supported after one pass, and the layer1_synthesis finalize exit fires. Note that the ledger entry itself records “late September 2025” while scoring the check supported, on a claim posted 13 January 2026. The empty arm stopped after one pass too, and its planner drew the same conclusion; its verdict was rescued by a judge holding four articles headlined *Border Patrol agents chase cyclist*, which its query phrasing had surfaced.

The cost has a population-level signature that mirrors the benefit exactly. An unresolved check does not only buy another pass; when the missing evidence cannot be found it drags the verdict to the middle of the scale. Pooled across both planners, the blueprint lands in the neutral band ($|s| < 1/3$) on 19.1% of claims whose ledger still holds an open check, against 4.4% when every check is closed, while ground truth lies in that band for only 0.2% of claims. On the 54 claims where the blueprint hedges and the empty arm commits, its MSE is 0.731 against 0.357. So one behavior yields three outcomes: where the evidence is reachable the open cell buys the pass that reaches it, where it is not the verdict is pulled somewhere the ground truth almost never

sits, and where the checks close on the wrong questions the graph stops early.

5.3. Where the Error Comes From

The main comparison leaves the size of the effect unexplained. This section attributes the error rather than measuring it again. Section 5.3.1 asks whether routing a claim to the right blueprint matters at all. Section 5.3.2 removes retrieval entirely and judges on the evidence a human fact-checker gathered, which bounds what any retrieval improvement could buy. Section 5.3.3 then asks which part of retrieval is responsible, separating pages the pipeline failed to fetch from sources it never put in front of the retriever.

5.3.1. What Routing Contributes

This section asks whether routing a claim to the right blueprint changes the verdict, and whether the selector routes reliably enough for that to matter.

Training Routing Fit. Whether the selector actually assigns the training claims to their corresponding blueprint depends on the selection method. Measured over all 3,655 clustered claims, LLM-based selection reaches a classification accuracy of 44%, while the hybrid approach (linear probe, with the LLM handling cases of low probe confidence) reaches 76.7%. The majority-class baseline of always choosing the largest blueprint is 54.3%, which the LLM alone falls short of. The two approaches ask different questions by design. The probe predicts provenance (*Which blueprint was built from claims like this?*), whereas the LLM-based approach reasons about fit (*Which of these investigative approaches suits this question?*). Both figures are in-sample, and every claim scored here also contributed to the clusters the blueprints were synthesized from, so they describe fit and say nothing yet about generalization.

Held-Out Routing Fit. A held-out split carries no cluster labels, since HDBSCAN labels only the points it was fit on. Reconstructing a routing target therefore means analyzing the held-out fact-check articles into the same strategy fingerprints used for training, projecting them through the UMAP reduction recorded in the clustering run, and assigning each to the blueprint its 15 nearest training articles belong to. Cross-validated on the training split, that rule recovers 95.2% of the correct blueprints, so the reconstructed targets carry roughly 5% label noise. Of the 1,000 claims in the 2026 Q1 split, 935 pass the process-richness filter that fingerprinting requires. On those claims the LLM reaches 52.6% and the hybrid approach 74.8%, against a majority-class baseline of 73.3%, while the probe alone reaches 83.9%. The held-out quarter is more media-heavy than the training data, which raises the baseline from 54.3% and makes the comparison to it the informative one. The hybrid’s accuracy barely moves (76.7% to 74.8%), yet its lift over the majority class collapses from 22.4 to 1.5 percentage points, as the LLM tie-break it defers to degrades under distribution shift. The probe shows no such degradation, matching in-sample accuracy on a class-balanced training sample of equal size (83.6%), which places its ceiling at the separability of blueprint provenance from claim text rather than at overfitting.

Effect of Routing Fit. Experiments along the way pointed the same direction, from the error side rather than the accuracy side. We forced most traffic onto the dominant blueprint, then forced it away. Each intervention rewrites a large share of the verdicts, yet the error stays where it was. Even claims that carry media survive assignment to a blueprint written for text. We read the two controls as a bound on what any router can contribute here, and we stopped investing in the mechanism.

Pool Size. Grouping claims by the investigation path a fact-checker took is natural on the training side, since the paths differ. From the selector’s side it is not, because a new claim carries no path yet and several blueprints may plausibly fit. Larger pools were consistently worse in development. Pool size moved together with the clustering configuration in those runs, so the comparison is confounded and we report it as the reason the pipeline consolidates aggressively rather than as a measurement of pool size alone. It does not contradict the routing controls above. Those hold the pool fixed and vary which blueprint runs, whereas pool size changes how separable the blueprints are in the first place.

5.3.2. Judging on Ground-Truth Evidence

A pipeline’s error has two possible sources: (1) the judge never saw the evidence that would settle the claim, or (2) it saw it and mapped it onto the wrong verdict. The two require opposite remedies, and an end-to-end score cannot separate them. We therefore remove the first source entirely, replacing retrieval with the evidence a human fact-checker actually gathered, and measure what remains.

Setup. Each claim in the benchmark is accompanied by the fact-checking article that adjudicated it. We extract from that article the set of source-attributed findings the fact-checker established, using an LLM (GEMINI 3 FLASH, $T=0$) constrained to report observations rather than conclusions. The fact-checker’s own rating, and each finding’s direction with respect to the claim, are recorded separately and withheld from the judge (an article states its verdict explicitly, and a direction label on a decisive finding restates it, so including either would measure verdict transcription rather than evidence quality). Extraction succeeds for 998 of 1000 claims, yielding 6334 findings (6.3 per claim; 3121 textual, 1947 contextual, 1266 authenticity). The two failures are scraped pages containing no fact-check content. For the 492 claims whose text is a corrected restatement of the debunked original, the original claim is supplied so findings are oriented against the claim as stated. The judge is then run on the claim and these findings alone, with no blueprint, planner, or retrieval, under the configuration used by the pipeline runs. If not stated otherwise, the experiments are run with GEMINI 3 FLASH using 3 samples. Manual inspection of 25 claims revealed no leakage to the judge.

The ceiling. Table 5.6 shows that ground-truth evidence reduces MSE by 85–88% relative to the blueprint pipeline for every judge except GEMMA 4 31B. The difference between small models such as QWEN 3.5 35B and large frontier models such as CLAUDE OPUS 5 is practically negligible. Paired over the 997 commonly scored claims, GEMINI 3 FLASH and QWEN 3.5 35B are statistically indistinguishable from both Opus generations in MSE, so given ground-truth evidence a large model is not worth the roughly $13\times$ cost. The residual error is one of calibration rather than of direction. For GEMINI 3 FLASH only 9 of 998 claims (0.9%) receive a verdict of the wrong sign, and the same-direction errors are symmetric (mean confidence bias -0.012), so no global rescaling would remove them; excluding the sign errors leaves MSE 0.024, meaning the 0.9% of claims that flip carry 46% of the total squared error. GEMMA 4 31B is instructive as the exception. It is the only markedly over-confident judge (bias $+0.080$), which simultaneously earns it the highest 7-class accuracy (71.11%, by

	MSE ($\downarrow$)	MAE ($\downarrow$)	Acc. 7 ($\uparrow$)	Acc. 3 ($\uparrow$)	Flips ($\downarrow$)	MSE excl. flips ($\downarrow$)
GEMINI 3 FLASH	0.045	0.130	61.1%	97.1%	0.9%	0.024
CLAUDE OPUS 5 [†]	0.042	0.128	65.0%	97.4%	1.0%	0.025
CLAUDE OPUS 4.6	0.044	0.126	66.2%	97.3%	1.1%	0.024
GEMMA 4 31B	0.094	0.139	71.1%	96.0%	2.6%	0.020
QWEN 3.5 35B	0.049	0.131	63.8%	96.1%	1.1%	0.027
QWEN 3.5 397B	0.047	0.123	64.9%	97.1%	1.2%	0.023
QWEN 3.6 35B [†]	0.049	0.117	68.5%	95.5%	1.3%	0.022

Table 5.6.: Comparison of different models as judge given ground-truth evidence using 3 samples. Flips defines how many times the judge confuses  **Intact** and  **Compromised** independent of confidence. MSE excl. flips excludes the flipped samples from the MSE computation. Best values are marked in **bold**. [†] Subject to data leakage as knowledge cutoff is after claim dates (see table 5.2)

committing to decisive labels) and the lowest calibration error of all judges once sign errors are excluded (0.020), while tripling its flip rate to 2.6%. Because a sign error costs up to 4.0 squared error on a $[-1, 1]$ scale, those 26 claims account for 79% of its total error. Accuracy and MSE therefore reward opposite behaviors under this metric, and GEMMA 4 31B’s inverted ranking across the two columns is a direct consequence rather than an anomaly.

The residual sign errors are largely shared and structural rather than random. Of the 22 claims flipped by at least one of the judges excluding GEMMA 4 31B, 5 are flipped by all six, and none lies near the sign boundary (mean $|\text{ground truth}| = 0.81$). They are concentrated in claims whose decisive property is context coverage (54.5% of flips against a 35.5% base rate; flip rate 3.39% versus 1.56% elsewhere), and among the five claims all six judges flip, the error is one-directional, i.e. the judge returns  **Intact** where the label is  **Compromised**. Inspection confirms the pattern that these are claims the fact-checker rates as correctly attributed or true, but scores as  **Compromised** because decisive context is omitted. Given ground-truth evidence stating that the quote was indeed uttered, concluding  **Intact** is the correct inference about veracity and the wrong one about integrity. The remaining error is thus of two kinds: a systematic under-weighting of context coverage in the judge’s rubric, and, for the $\sim 99\%$ of claims judged in the right direction, calibration of certainty.

What retrieval misses. To attribute the remaining gap, we score each pipeline run’s evidence against the gold findings. An LLM decides, for every gold finding, whether the fact it states appears anywhere in what the pipeline retrieved. Because findings carry an integrity-component label from extraction, recall can be broken down by component. Our comparison covers a blueprint run and a single-strategy run with an empty strategy, matched in configuration wherever the two architectures allow. Finding recall is near-identical across the two, at 48.3% for the blueprint run and 49.4% for the single-strategy run. Both are uneven across components, and uneven in the same order. Textual findings reach 55.8% and 58.3%, contextualization 46.6% and 44.9%, and media authenticity only 32.7% and 34.9%. Absolute values shift by up to 5 percentage points with the scoring prompt, though the ordering across components is stable and reproduces in both runs. Scoring the pipeline evidence against gold instead shows that the retrieved evidence is diffuse. Only 9.8% and 7.7% of items are off-topic, and 1.4% and 1.3% contradict a gold finding. A larger share is simply un retrievable, since 18.7% and 18.9% of all evidence items are pages that could not be fetched and reach the judge as bare search snippets. That this figure holds across two different architectures points to the scraping layer rather than to either planner. Recall predicts pipeline error strongly in both runs. Table 5.7 shows that sorting claims into recall quartiles roughly triples pipeline MSE from the highest quartile to the lowest ($3.3\times$ and $2.9\times$). Claim

	Blueprint		Empty Strategy	
Recall quartile	Pipeline MSE ($\downarrow$)	Gold MSE ($\downarrow$)	Pipeline MSE ($\downarrow$)	Gold MSE ($\downarrow$)
Lowest recall	0.503	0.023	0.497	0.023
Second	0.384	0.034	0.264	0.029
Third	0.182	0.065	0.173	0.049
Highest recall	0.151	0.057	0.173	0.081

Table 5.7.: Pipeline error against retrieval recall, by quartile. Claims are sorted by the fraction of ground-truth findings the pipeline recovered, then split into quartiles. *Pipeline MSE* is the run’s own error on those claims, while *Gold MSE* is the error the same claims incur when the judge receives ground-truth evidence instead of retrieved evidence. Pipeline error falls by a factor of three as recall rises, whereas the gold-evidence error stays well below it in every quartile and shows no comparable structure, so the effect is attributable to what retrieval delivered rather than to the claims being harder. The gold column’s own spread rests on very few claims: the gold-evidence judge reverses direction on 0, 1, 2, and 6 of the 240 claims per quartile, and excluding those the values are 0.023, 0.023, 0.025, and 0.027.

difficulty does not explain this. Judged on ground-truth evidence, the same claims score between 57.89% and 64.78% accuracy across every quartile of both runs, so the claims retrieval fails on are no harder to adjudicate. They are simply the ones where retrieval found least. The quartiles are also balanced on label composition, mean |ground truth|, gold findings per claim, and pipeline evidence volume.

Filtering versus supplying what is missing. Given that the retrieved evidence is diffuse, we ask whether removing the surplus helps, and whether supplying the missing findings does. All arms re-judge the same claims with the same judge, and are reported on the 743 claims every arm scores. *Oracle* filtering keeps only items matching a gold finding and cannot run at inference time; *relevance* filtering drops off-topic items and unfetchable stubs and could be learned; *random* filtering is matched to oracle per claim in item count, and is required because removing evidence also shortens the prompt. The augmentation arms add gold findings the pipeline missed, findings it already had (a volume control adding no information), or all of them. Table 5.8 shows the results and that filtering does not reliably help. The oracle and relevance gains on the blueprint run are not significant, and on the strategy run both filters significantly *hurt* because they remove relevant evidence faster than noise. Random filtering is severely damaging in both runs, which is why the volume-matched control is necessary as without it the oracle result would read as a filtering effect rather than a length effect. Supplying the missing findings is the only intervention that moves the number with -50% and -52% MSE. Its volume control, however, accounts for a large share of that. Re-stating findings the pipeline already had, adding no information, yields -24% and -20% MSE. Finally, adding *all* gold findings to the pipeline’s own evidence still lands well short of gold alone (0.090 and 0.088 versus 0.052) which means the pipeline’s surplus evidence measurably dilutes a clean evidence set even when nothing is missing from it.

5.3.3. What Retrieval Failure Costs

The agent reads a web page by fetching it and summarizing what it finds. Fetching fails often enough to matter, and the failures are not spread evenly over the web. Paywalls, bot defenses, and interstitials concentrate them on exactly the kind of established news outlet a fact-checker is likely to cite. That raises the question this

Evidence given to the judge	Blueprint			Single-Strategy		
	MSE ($\downarrow$)	Acc. 7 ($\uparrow$)	Acc. 3 ($\uparrow$)	MSE ($\downarrow$)	Acc. 7 ($\uparrow$)	Acc. 3 ($\uparrow$)
Pipeline evidence, unfiltered	0.270	42.5%	86.8%	0.203	46.3%	86.9%
filtered to gold matches [†]	0.220	41.6%	85.1%	0.217	43.3%	84.7%
filtered: drop off-topic & stubs	0.248	40.5%	84.9%	0.208	42.4%	86.9%
filtered at random, volume-matched	0.355	36.2%	74.7%	0.305	36.2%	76.7%
+ gold findings already held	0.207	42.1%	89.2%	0.163	47.6%	90.2%
+ gold findings missed [†]	0.134	49.0%	93.5%	0.097	52.8%	94.4%
+ all gold findings	0.090	50.9%	94.6%	0.088	54.0%	95.4%
Ground-truth evidence only (ceiling)	0.052	61.6%	96.4%	0.052	61.6%	96.4%

Table 5.8.: Evidence ablations for both pipeline variants, on the 743 claims every arm can score. The first block filters the pipeline’s own evidence, the second augments it with findings extracted from the fact-checking article. [†]Marked arms consult the ground truth to decide what to keep or add and are upper bounds, not deployable configurations; they also account for essentially all of the restriction to 743 claims, since keeping only gold-matching evidence leaves nothing to judge where retrieval recovered none. That restriction is conservative with respect to our conclusion: on the excluded claims the unfiltered pipeline is worse (0.385 MSE) and ground-truth evidence better (0.024), so the full set would show a larger gap. The ceiling row is identical across variants because it uses no retrieved evidence. The unfiltered row re-judges the run’s own evidence under the ablation harness so that every arm shares identical judging conditions; the run’s own figure on these claims is 0.278.

Site	Retrieval in this run			Fact-check articles	
	Attempts	Failed	%	Claims citing	Harmed
reuters.com	585	272	46	38	13
afp.com	441	207	47	8	4
nytimes.com	243	123	51	22	10
ndtv.com	172	81	47	5	1
timesofisrael.com	132	70	53	7	2
france24.com	109	56	51	25	5
congress.gov	96	38	40	6	2
researchgate.net	72	37	51	7	3
war.gov	59	29	49	2	0
washingtonpost.com	52	29	56	3	0

Table 5.9.: The ten sites absorbing the most retrieval failures, over 1000 claims. *Claims citing* is the number of claims whose fact-check article credits the site at all; *harmed* is the subset where the pipeline attempted that site for that claim and the fetch failed. A high failure rate is only costly where the two right-hand columns are also large, which holds for Reuters and the New York Times and not for the government domains.

section answers. Of the pages we cannot retrieve, how many were the ones professional fact-checkers actually used to reach their verdict?

Outcome	Self-citations held out		Self-citations counted	
	<i>n</i>	% (95% CI)	<i>n</i>	%
Scraped successfully	438	17.1 [15.7, 18.7]	504	15.1
Attempted and failed	112	4.4 [3.6, 5.2]	204	6.1
Never attempted	2004	78.5 [76.8, 80.1]	2632	78.8
(claim, site) pairs	2554		3340	

Table 5.10.: What retrieval did with each source the fact-checkers used, over 1000 claims. Intervals are a 2000-replicate bootstrap resampling claims, which is the unit of independence; attempts within a claim are correlated. The right-hand columns fold back the unlinked references to the claim’s own outlet, showing that the ordering does not depend on that judgment.

Measuring failure. Each time a worker agent completes a delegation, it records the sources it tried to read and any errors it encountered. A retrieval attempt is one (claim, task, URL) triple, and it counts as a failure if the retriever returns no content and leaves only the search-engine snippet behind. URLs are reduced to their registered domain, so `abc.com/x/y` and `abc.com/z` are considered to be the same site. Over the 1000-claim run analyzed here, 2536 of 25,609 attempts failed, a rate of 9.9%. However, one bias runs through this denominator. A page that is fetched successfully but yields nothing relevant leaves no record of its own, so successes are undercounted and per-site rates are mild overestimates. Failure counts are exact. Table 5.9 lists the sites that absorb the most failures. The pattern is a paywall pattern. Established news outlets fail close to half of the time, whereas social platforms, which were once more challenging, now rarely fail. Instagram accounts for 543 attempts with a 5% failure rate, but does not appear in the table.

Recovering the sources fact-checkers used. The comparison set comes from the original fact-check articles. For each claim we take the sources its article credits for a finding, which excludes the navigation, sharing, and related-coverage links filling much of a published page. Of the 6334 credited sources, 27.4% are linked and resolve to a site directly. The rest are named in prose, as *AFP México* or *U.S. House of Representatives’ official directory*, and are matched by name against the sites the articles link. Together these routes resolve 51.5% of credited sources. A further 31.1% are held out, being either instruments such as reverse image search, which retrieval was never meant to fetch, or unlinked references to the article’s own outlet, which mark the checker’s own reasoning more often than a retrievable page. The remaining 17.3% are named individuals and one-off documents with no site by nature. Throughout, the claim’s own fact-check article is excluded by the same comparison the pipeline’s leakage guard applies to it, which removes 5 sources.

Results. Each resolved source is placed in one of three buckets, per claim, according to what retrieval did with that site while investigating that claim. It was fetched, it was attempted and failed, or it was never attempted at all. Table 5.10 gives the split. Retrieval failure accounts for 4.4% of the sources fact-checkers used, and sources the pipeline never attempted for 78.5%. Counting unlinked self-citations as retrievable moves the failure share to 6.1% and leaves the never-attempted share at 78.8%, so the ordering does not rest on that judgment.

Failures are, however, mildly enriched for relevance. Sites cited by some fact-check article in the dataset fail at 21.0%, against 16.0% for all other sites, a ratio of 1.31. The enrichment is real and small. It reflects the paywall pattern of Table 5.9, since the outlets that defend themselves against automated reading are also the outlets professional fact-checkers cite.

Search queries fare no better. In the same run, 29.5% of queries returned no sources at all. Exact-phrase quoting accounts for most of that with quoted queries coming back empty 69.8% of the time against 12.6% for the rest, and the gap holds both within single claims that issued each kind (69.8% against 14.7%) and within each writing script, so it follows the syntax rather than the difficulty of the claim. The planner quotes claim text it has only seen translated or paraphrased, which no page carries verbatim. Warning it off the practice and retrying every empty quoted query without its quotes halves the waste from 30.3% to 13.7% of queries on a paired 200-claim rerun, and changes nothing downstream. Finding recall moves from 49.7% to 49.0%, and the never-attempted share is unmoved. Query yield is not what separates the pipeline from those sources.

Interpretation. Perfect retrieval would recover about one in twenty-three of the sources behind the reference verdicts. Four in five are lost earlier, at a stage where the pipeline never puts the source in front of the retriever at all. Reuters illustrates both halves. It is the most-attempted site in the run and fails 46% of the time, yet fact-check articles credit it for 38 claims and only 13 of those are claims where a Reuters fetch was attempted and failed. Hardening the fetcher against paywalls is worth doing and would move the smaller number. Whatever governs which sources are surfaced and selected governs the larger one, and this thesis does not isolate that stage. Section 5.3.2 bounds what it costs, since supplying the findings the pipeline never retrieved is the only intervention that moves the error.

5.4. Ablations

The preceding section asks where the error comes from. The two studies here ask a narrower question, whether a specific addition to the pipeline reduces it. Both return a null, and both are reported because the component was plausible enough to build and because each bounds its own upper limit, one by a placebo arm and one by an oracle.

5.4.1. Refining with unused Budget

The execution graph of a blueprint is an acyclic tree where different paths can have different lengths. The budget a planner has is at least as many iterations as the length of the longest path, which in turn means that executions taking a shorter path do not get to use the full budget. In practice the slack is larger still and runs use only 3.23 of 8.5 allowed iterations on average. Termination is instead caused by the planner finalizing voluntarily (68.1% of claims) or by the graph running out of nodes (31.7%), the latter group ending with a mean of 1.25 of its activated checks still unresolved. We ablate whether using this budget in a refine step before finalizing enhances performance.

Setup. A synthetic *refine* node is added that is entered whenever the graph would finalize while (i) iteration budget remains and (ii) at least one check activated on the taken path is still unresolved. The node is not declared in any blueprint, it is injected at routing time and applies uniformly. It may re-enter itself to spend the remaining budget, and is bounded by the iteration limit and by the planner’s own decision to finalize, which its prompt explicitly invites when further search cannot change the verdict. The prompt separates unresolved checks on the taken path, which are the targets, from checks belonging to branches the claim never entered, which are offered as non-binding suggestions only. Check activation is path-scoped precisely because a branch’s checks may be unanswerable for a claim that did not take it. To isolate the effect from

sampling noise, we do not re-run the pipeline. We instead reconstruct each claim’s state at the moment its graph finalized (evidence, check ledger, node and delegation history) from the recorded traces, and resume execution into the refine node. The pre-finalize trajectory is thus held fixed and only the refine iterations vary. Of 1000 claims, 287 (28.7%) meet the entry condition and the remaining 713 are unaffected by construction and are carried over unchanged. Judge configuration is identical to the base run (GEMINI 3 FLASH, 3 samples).

Results. Refine does not improve performance. Over all claims, MSE moves from 0.307 to 0.301 and 7-class accuracy from 30.1% to 30.6%. 21 claims are corrected and 16 are broken. Restricting to the 286 resumed claims (one failed) gives $0.419 \rightarrow 0.398$ MSE. The resumed set contains its own control. On 129 of the 286 claims the planner declined to delegate and finalized immediately, so these were re-judged on byte-identical evidence and isolate the effect of judge resampling alone. Their MSE improves by $\Delta\text{MSE} = -0.018$ ($0.295 \rightarrow 0.277$), against -0.0226 ($0.520 \rightarrow 0.498$) for the 157 claims where refine did gather new evidence. The difference-in-differences attributable to the additional evidence is therefore only -0.005 , i.e. roughly four fifths of the apparent gain is resampling. We conclude that the unspent budget is not a missed opportunity and the planner stops early because further search on the same claim yields more evidence of the same kind rather than the evidence that is missing. This is consistent with our other ablations, in which supplying the specific findings the pipeline failed to retrieve reduces MSE by 50%, while adding volume without new information accounts for a large share of any measured gain.

5.4.2. Does Manipulation Detection Help?

The detector comparison in Appendix A.3.2 establishes how well the available tools separate manipulated from authentic files, but detector accuracy is not the quantity of interest. The question is whether adding such a tool changes the pipeline’s final verdict. A detector can be accurate and still be worthless here, either because the pipeline already reaches the same conclusion by other means or because file integrity is not what the claim turns on.

Setup. Three arms are compared on identical claims and an otherwise identical configuration, differing only in which detector the media agent is given: none (the baseline), Sightengine, and an *oracle* detector that returns the ground-truth integrity label and therefore represents perfect detection. The oracle returns only a verdict and a manipulation type; the fact-check’s own justification, which names the manipulation and effectively contains the answer, is never exposed, since revealing it would measure a perfect researcher rather than a perfect detector. It remains a generous upper bound, because its labels derive from the same fact-check text that produced the gold veracity label.

The sample is the 277 claims of the 2026 Q1 split whose image carries a non-unknown integrity label and that contain no video, matching the images-only scope of the detector evaluation; 274 were scored successfully in all arms and are compared paired. Because a null result could equally mean that the verdict stage is too weak to exploit the added evidence, every arm is additionally re-judged by a frontier model (CLAUDE OPUS 5) over the identical stored evidence, with only the judge changed.

Results. Under the GEMINI 3 FLASH judge, neither detector improves anything. The baseline is nominally best on every metric, no paired difference in MSE is significant, and both arms leave roughly two thirds of the verdicts untouched. This holds for the oracle as well, i.e. even perfect knowledge of whether the file was

Arm	GEMINI 3 FLASH			CLAUDE OPUS 5		
	MSE	Acc. 7	Acc. 3	MSE	Acc. 7	Acc. 3
No detector (baseline)	0.193	38.3%	92.7%	0.149	42.0%	93.8%
Sightengine	0.217	37.6%	92.3%	0.171	41.2%	93.4%
Oracle (perfect)	0.198	35.8%	92.0%	0.145	48.5%	94.2%

Table 5.11.: Manipulation detection ablation, paired over $n = 274$ claims, under two verdict stages. Only the judge model differs between the two halves; the investigations, and therefore the evidence, are identical. Best value per judge in bold.

Manipulation type	n	% comp.	GEMINI 3 FLASH			CLAUDE OPUS 5		
			None	Sight.	Oracle	None	Sight.	Oracle
AI-generated	90	38.9	0.206	0.149	0.153	0.137	0.133	0.091
Graphic edit	34	44.1	0.127	0.222	0.147	0.105	0.232	0.170
Fabricated screenshot	33	51.5	0.057	0.054	0.040	0.094	0.037	0.030
Splice / composite	14	42.9	0.063	0.071	0.111	0.056	0.056	0.048
Deepfake	11	45.5	0.323	0.566	0.091	0.091	0.263	0.051
Other [†]	1	100.0	2.778	2.778	2.778	0.111	0.111	0.111
Authentic (no manipulation)	91	39.6	0.228	0.293	0.315	0.219	0.242	0.256
All	274	42.0	0.193	0.217	0.197	0.149	0.171	0.145

Table 5.12.: MSE by the ground-truth manipulation type of the claim’s image, over the same paired $n = 274$ claims and the same three arms as Table 5.11. % comp. is the share of the group whose gold verdict is compromised, given for reference against the 42.0% base rate. Lowest MSE per group and judge in bold. Every group is a post-hoc split and the small ones are severely underpowered, so the entries are observations rather than findings. [†]One claim, shown so that the rows account for the whole sample. All three arms return the same verdict on it under both judges, so it contributes nothing to any comparison, and its large GEMINI 3 FLASH entry is that judge reversing the direction with confidence on a single claim.

altered does not produce a better verdict. The null is not an artifact of the tool going unused, since the planner invoked the authenticity check on about three quarters of the claims and restricting the comparison to those does not change it.

Under the CLAUDE OPUS 5 judge every arm improves substantially, but the conclusion survives with one qualification. Sightengine remains indistinguishable from the baseline. The oracle remains null on MSE and on coarse three-class direction, yet gains 6.6 percentage points of exact-label accuracy. That combination is coherent rather than contradictory as perfect knowledge of file integrity sharpens the *certainty* attached to a verdict without changing its *direction*. None of that narrow benefit is accessible to a real detector because the oracle outperforms Sightengine on nearly twice as many claims as the reverse, which is the only unambiguously significant comparison in the ablation and measures the gap between perfect and achievable detection rather than any gain over the baseline.

Which manipulations, if any, benefit. Table 5.12 splits the same 274 claims by the kind of alteration their image carries. A claim inherits the type of its manipulated image, and the three claims whose images carry two different types are resolved toward the more specific one. Under CLAUDE OPUS 5 the oracle records the lowest error of the three arms on every manipulated category except graphic edits. On authentic images the ordering inverts under both judges, and there the oracle is the worst of the three. Knowing that a file was altered narrows the space of readings, whereas being told the file is genuine appears to license trust the claim has not earned. Aggregated over the integrity split, the oracle gains 0.036 MSE on manipulated images under GEMINI 3 FLASH and loses 0.087 on authentic ones, which is how a per-category effect of either sign cancels to nothing overall. None of the individual rows should be read as a finding. Five of the seven groups hold fewer than thirty-five claims, and in the smallest of them fewer than ten verdicts differ from the baseline at all, so one claim can reorder a row. The oracle’s large gain on deepfakes comes from four claims that moved.

Interpretation. Establishing that a file is an unaltered capture does not help judge a claim that is false for a different reason (a genuine photograph presented with a false caption, date, or location) and the added evidence perturbs the verdict about as often as it anchors it. On this corpus the pipeline’s difficulty is not determining whether an image was manipulated but determining what it depicts and whether the claim’s account of it holds, and that question is answered by provenance evidence rather than by pixel forensics. Two magnitudes frame the decision. The oracle bounds what any detector could contribute, and that bound is null on the primary metric under both judges, while changing the judge model alone improves the baseline by 0.044 MSE which is an order of magnitude more than any detector effect measured here. Our results coincide with the results of Sagar et al. who found that pixel-level-only deepfake detectors perform weakly, and adding their outputs to fact-checking often hurts performance [111]. Manipulation detection is therefore not enabled in the main experiments. The tools remain implemented and can be switched on, but no configuration of them is justified by these results.

Three limitations qualify this. The sample covers image-only claims from a single split and the confidence intervals on the oracle comparison admit a true effect of up to roughly ± 0.05 MSE in either direction. The claim is that no benefit worth the dependency is detectable at this sample size, not that the benefit is exactly zero.

6. Discussion and Future Work

Chapter 5 measured the pipeline. This chapter reads those measurements. It opens with a direct answer to each of the three questions of Chapter 1, stated together with what it does not cover. The rest of its length goes to the second and third, since a negative result is useful only once its cause is named. Section 6.2 traces the blueprint reversal to a single property of the design. Section 6.3 asks what the error that remains is worth, and Section 6.4 asks why procedural guidance does not reach it. Section 6.5 turns to what the measurement decisions of Chapter 5 imply for evaluating systems of this kind, before the limitations of the work and the openings it leaves.

6.1. Answers to the Research Questions

Can verification strategies be learned from published fact-checks? They can, and no manual annotation of the investigative process is required anywhere. Of the 4,000 training claims, 291 report a verdict without recording how it was reached and are dropped for that reason. The remaining 3,709 fingerprints cluster into 14 groups covering 98.5% of them, each of which synthesis turns into a blueprint. Claim text alone then predicts which of those procedures a fact-checker followed, at 83.9% on the held-out quarter against a 73.3% majority baseline. The probe scores 83.6% in sample, so that figure reflects how separable blueprint provenance is rather than how well the probe fits its training data (Section 5.3.1). One qualification limits how much of the recovered diversity reaches the runtime. Consolidation compresses the 14 clusters to four blueprints plus a generic fallback, and two of those four absorb 90% of clustered traffic, so the executed pool is far coarser than the corpus it was induced from. A second qualification is structural. Blueprints encode what fact-checkers did, which is a strict superset of what the agent can run, and Section 6.6 takes up what that costs.

Does conditioning verification on a per-claim strategy improve accuracy? Not with the representation used here, and the sign of the effect depends further on which planner executes it. Averaged over five judges, learned blueprints improve MSE for the GEMMA 4 31B planner from 0.401 to 0.347 and degrade it for the GEMINI 3 FLASH planner from 0.318 to 0.358. Both movements are small beside the spread that the choice of judge introduces. Unanimity carries them, since all five judges rank the conditions the same way under each planner. Replacing the routed pool with one distilled document recovers 0.042 of the 0.054 gain on GEMMA 4 31B, which locates most of the benefit in the guidance rather than in the per-claim routing of it. Section 6.2 takes up why a better selector does not recover it, since what would make one blueprint fit is only revealed once a run is under way. The comparison that settles the question is simpler than either contrast. GEMMA 4 31B with blueprints reaches 0.347 and GEMINI 3 FLASH with no guidance at all reaches 0.318, so the routed pool converges the two planners at a level the stronger one already beat unaided. Guidance of this form appears to substitute for a planner's own procedure instead of adding to it, which suggests that the effect shrinks further as planners improve. The scope of the answer should be stated plainly. The grid tests

one representation, one pool of four executed blueprints, and one router, on one benchmark of one quarter. Different graph semantics, or conditioning at the level of individual checks rather than whole procedures, may well behave differently, and nothing measured here rules that out. Pool size is the one variant we have any evidence on, and it points the other way. Larger pools were consistently worse in development, though pool size moved together with the clustering configuration in those runs, so that observation is confounded rather than decisive (Section 5.3.1).

If it does not, where does the error actually come from? In this pipeline the error comes from retrieval, and within retrieval from the stage that decides which sources to consult. Three measurements narrow it down in sequence. Judging on the evidence a human fact-checker gathered cuts MSE by 85 to 88% and leaves 0.9% of claims judged in the wrong direction. A 35B judge is statistically indistinguishable from a frontier model there, so the verdict stage is close to saturated and scaling it buys little (Section 5.3.2). Claims sorted by how much of that evidence the pipeline recovered then show three times the MSE in the lowest recall quartile as in the highest. The same claims judged on ground-truth evidence show no comparable structure, which attributes the error to what retrieval delivered rather than to the claims being harder. A split of retrieval itself puts 4.4% of the sources fact-checkers used in the fetch-failure bucket and 78.5% in a bucket the pipeline never attempted at all (Section 5.3.3). Two further results agree with that reading. Halving the pipeline’s wasted queries moves finding recall from 49.7% to 49.0%. Adding every gold finding to the pipeline’s own evidence still lands at 0.090 MSE against 0.052 for the gold findings alone, so surplus evidence dilutes a clean set. More retrieval does not fix this, and cleaning up the queries that produce it does not either. The stage that governs which sources are surfaced and selected is the one that binds, and it is the stage this thesis does not isolate.

Taken together the three answers point to where effort in this design should go. Verification procedure is recoverable from the published record and executable at inference time. Conditioning on it in the form built here left accuracy unchanged at best, and the error that remains is dominated by a stage upstream of everything the strategy layer controls.

6.2. Why the Blueprint Effect Reverses

Section 5.2 traces the reversal to a single property of the blueprint condition. Its required checks are separate cells, so a fact that closes one of them closes only that one, and no baseline carries that structure. Three outcomes follow from that property alone. Where the missing evidence is reachable, an open cell buys the retrieval pass that finds it. Where it is not, the same cell pulls the verdict into a band that holds 0.2% of the ground truth. Where the checks ask the wrong question, they close and the graph exits early (Figures 5.5 and 5.6). Neither retrieval quality nor execution budget separates the arms, and the blueprint condition is the better-resourced one wherever it loses.

That property is symmetric across the two planners, and the reversal is not. Both pay the hedging cost, at +0.025 and +0.014 MSE on the claims where only the blueprint hedges, so what differs between them is the size of the benefit rather than the size of the tax. A blueprint supplies a procedure the planner would otherwise supply itself. A planner with a weaker procedure of its own therefore gains more from being told what must be established than it loses from being held to it, which is the substitution reading of Section 6.1. On that reading the value of a learned procedure falls as the planner’s own competence rises. The axis is planner strength and not the presence of a reasoning trace, since both planners here reason step by step. Two

planners are two points on that axis, so the reading is a direction the grid supports rather than a slope it measures.

Neither the graph nor the per-claim routing accounts for what value there is. Forcing traffic onto the dominant blueprint, and forcing it off, each leaves the error where it was, while the distilled document reaches most of the same gain carrying no graph, no ledger, and no selection at all. That routing does not move the error is not a shortfall of the selector. Provenance is separable from claim text, and the probe recovers it at 83.9%. Recovering which blueprint a claim resembles is not the same as knowing which one yields the correct verdict. A blueprint is induced from the investigation path a fact-checker followed. At run time only the claim exists. The path that would make one blueprint fit better than another is produced by executing it, not read off the claim. Several blueprints plausibly fit a given claim, and the quantity a router would need to rank them is only revealed once a run is under way. A stronger selector cannot close this gap, because the information it would need does not exist at the point of selection. Beyond the distilled document, the blueprint condition adds the ledger, which is also the component that costs.

The actionable conclusion is narrower than the negative result. Our ledger binds two quantities that have no reason to move together, namely whether a check is settled and how firmly the verdict should be stated. An unresolved check is a good reason to keep searching, and a poor reason to report one-sided evidence as uncertain. Since both halves of the effect trace to that coupling, separating the two is the change this analysis most directly supports.

6.3. What the Remaining Error Is Worth

The ceiling result of Section 6.1 locates the error, and this section asks what it is worth. Its two halves respond to capability in opposite ways. Swapping a 35B judge for a frontier model changes nothing once the evidence is right, while swapping the evidence under a fixed judge moves the error by nearly an order of magnitude. The distance between those two numbers is the headroom this design still holds, and none of it is reachable by improving the stage that produces the verdict. That distance is also larger than any architectural effect this thesis measures. Blueprints move MSE by at most 0.054, the choice of judge by up to 0.14 within a single cell, and the evidence a claim is judged on by roughly 0.25. Which evidence reaches the judge therefore matters more than which model judges it, and both matter more than the procedure the planner was told to follow.

Within retrieval,^l the loss is concentrated in a stage the pipeline never makes explicit. Four in five of the sources fact-checkers credited were never put in front of the fetcher, against one in twenty-three lost to fetch failure. No component owns the decision of which sources a claim deserves. It falls out of which pages a search engine happens to surface and which snippets a planner happens to pick, so every intervention tried here operates downstream of it and none of them reaches it. Hardening query syntax halves the wasted queries and moves recall the wrong way, spending the unused iteration budget buys almost nothing, and a stronger judge changes nothing about what it was handed. A stage with no component is also a stage with no ablation, which is how it stayed invisible while the components around it were measured one by one.

Volume is not the remedy either. Surplus material measurably dilutes a clean evidence set, since supplying every gold finding on top of the pipeline's own still lands well short of those findings alone, and the one filter that could run at inference time hurts on a second run. Re-stating findings the pipeline already held cuts error by a fifth to a quarter while adding nothing it did not have, so part of any gain measured from more evidence is a gain from more text. The analysis points to a smaller and better-targeted evidence set rather than a larger one, which makes this a sourcing problem rather than a coverage problem. Improving how

sources are chosen is the only intervention available in this design that can still move the number, and every other stage is bounded by what that one delivers.

This diagnosis is cheap to reproduce and worth reporting as a metric in its own right. Any benchmark built from published fact-checks carries the sources its checkers credited, so recall against them can be computed for any system without new annotation. Reporting it alongside the end-to-end score separates a system that reasons poorly from one that never saw the page, which an aggregate verdict metric cannot do. The estimate here is bounded by site-level matching and a single run, and it is precise enough to locate the stage without being precise enough to score it.

6.4. Why Procedural Guidance Misses Sourcing

Locating the constraint in sourcing raises an obvious objection to the approach itself. Telling a planner which evidence settles a claim is precisely what a verification procedure is for, so guidance should show up as better evidence. It does not. The blueprint and single-strategy runs recover 48.3% and 49.4% of the findings fact-checkers established, and the same ordering across evidence components appears in both. Three properties of the design explain why.

The first is what a blueprint can name. Strategy fingerprints are topic-stripped so that a procedure generalizes beyond the claim it was induced from, which leaves the synthesized guidance able to name classes of source and never an instance. The record blueprint asks for the official gazette, the statistical office, or the peer-reviewed literature, and the never-attempted loss occurs one step below that. Turning “the relevant statistical office” into the page that carries this figure is query formulation and result ranking, which no field of the representation addresses.

The second is where the guidance applies. Per delegated task the retrieval budget is identical in every arm, at two search rounds of three queries keeping three results each. A blueprint therefore changes how many tasks are delegated and when the planner stops, and it cannot change what any one search returns or which returned result is fetched. Its action space is control flow, and the constraint sits inside the retrieval loop that control flow calls.

The third is executability. Expert consultation decided 25.1% of the training claims and official records 26.2%, while the agent can search the web and analyze media. A blueprint that faithfully encodes “ask the institution directly” yields a delegation the runtime can only answer by searching for someone else who already did.

None of this refutes procedural guidance. It says the procedures learned here act on the stage that was already adequate and stop short of the stage that binds, which is a property of the representation rather than of the idea. Guidance that carries source targets would test the idea properly, and the material for it already exists, since the article analysis records which sources settled each claim and a quarter of them resolve to a site directly.

6.5. What Each Metric Needs Beside It

One theme runs through the measurement decisions of Chapter 5. No single number this pipeline produces can be read on its own, and each of the four below earned its keep by supplying the figure that makes another one interpretable. A method delta needs the judge spread beside it, since judges span 0.14 MSE within one

cell here while the methods under study move at most 0.054, which is why the blueprint result is reported as five judges agreeing rather than as a difference (Section 5.2). A headline metric needs a confidence-invariant companion such as AUC, because a hedging policy on its own decides whether a system wins squared error or exact-label accuracy (Section 5.2.1). An aggregate error needs its direction-flip rate, given that under one percent of claims can carry nearly half of it (Section 5.3.2). A component’s measured gain needs a ceiling and a null, which is how an oracle detector retired manipulation forensics and how a control arm re-judging unchanged evidence exposed four fifths of a refinement gain as judge resampling (Section 5.4).

The comparison against the published VERITAS baseline is the case where those companions carry the whole reading. Our pipeline wins MSE against every judge and loses both accuracy columns against every judge, and the split is not a disagreement about evidence. It is a disagreement about how confidently to report, since our runs hedge where the baselines commit and the benchmark coarsens a hedged step into **Unknown** even where the direction is right. AUC settles which reading is correct, because it is invariant to that policy and separates the systems in our favor. The direction-flip rate then adds what no aggregate shows, namely that every configuration reported here is more reliable on direction than every baseline, and that the margin comes from three-sample judge aggregation rather than from the media tools and larger retrieval budget the comparison is otherwise credited with. A system can therefore be scored as clearly better and clearly worse than a baseline on the same predictions, which is a property of the reporting policy and not of the pipeline underneath it.

6.6. Limitations

Run count and split coverage. Each cell of the main grid is one end-to-end run over 1000 claims. Re-executing the pipeline is expensive enough that differences are read against a measured resampling floor rather than against replicated means, which is weaker than repeated runs would be. Cost also fixes the evaluation to a single quarter, since six end-to-end runs and five rescorings of each already exhaust the budget available here, and extending the comparison to 2026 Q2 needs no methodological change beyond paying for it. Retrieval runs through third-party search and scraping services and over the live web, none of it pinned to a version, so a later replication may land at different absolute error levels than those reported here. Using one benchmark follows from the knowledge cutoffs of the models under test. Established fact-checking benchmarks predate them, so a verdict measured on one can be recalled instead of retrieved, and VERITAS is the only benchmark publishing splits dated after the models were trained. Two planner models make the capability axis a two-point comparison, which supports the direction of the interaction without constraining its shape.

The diagnosis rests on model-built instruments. The claim that retrieval sourcing binds depends on three artifacts that are themselves model outputs, namely the findings extracted from each fact-check article, the matcher deciding whether the pipeline retrieved each one, and the judge that scores the result. Recall shifts by up to five percentage points with the scoring prompt, so its absolute level should be read as an ordering across components and runs rather than as a measurement of coverage. Source attribution is coarser still, since matching runs at the level of the registered domain and cannot tell the page a fact-checker read from any other page on that site. A success there means the pipeline reached the site. Two judges that appear only in the ceiling experiments are also not provably unexposed to the evaluation quarter, one by a disclosed cutoff after it and one by a release date after it. Neither plans, retrieves, or scores a main-results run, and the exposed one lands within 0.003 MSE of an unexposed model on the same task, which bounds any memorization advantage well below the effects it is used to measure.

What the main comparison prices. The blueprint condition differs from its baselines in several respects at once, carrying a graph, a check ledger, per-claim routing, and an iteration budget of four to nine against the two planning rounds its baselines receive. Introducing the single-strategy arm separates the distilled content from the rest of that package, and no arm separates the ledger from the graph that holds it, which is the component Section 6.2 holds responsible. Consolidation further leaves four blueprints carrying traffic and two of them absorbing 90%, so what the grid tests is coarse procedural conditioning. Held-out routing accuracy is measured against reconstructed targets that carry roughly five percent label noise.

Tools the runtime does not have. The delegated tasks routinely ask for analyses the media agent cannot run. It offers exactly two intents here, reverse image search and geolocation; `assess_authenticity` exists but is disabled, following the manipulation-detection ablation. Over the run, 586 of 1,117 media delegations ask for at least one capability the agent does not offer, and the categories overlap because a single delegation can request several at once. 552 request the disabled detectors, a configuration decision, while 91 delegations across 69 claims, almost all carrying video, ask for audio, dubbing, lip-sync or frame-by-frame analysis that exists nowhere in the pipeline. Claim 103463 (Figure A.1) is the sharp case because the claim is entirely about audio, the pipeline never examines a waveform, and the blueprint reaches the right verdict only by routing around the gap to a published fact-check that did the listening. The mismatch is predictable, since blueprints are induced from fact-checking articles and therefore encode a procedure that is a strict superset of what the agent can run.

Distance from deployment. This evaluation measures per-claim error on a benchmark, and says nothing about the error rate a working fact-checker could use. Every configuration reported here reverses direction on roughly one claim in twelve, which rules out publishing a verdict without review, and the review effort that would make such output worth having goes unmeasured. The same holds for the assistive mode these numbers more plausibly support, where the system gathers evidence and ranks claims for attention while a person rules on them. No human baseline is run on the same claims either, so the scores here carry no expert anchor. Prompts, tool descriptions, and the judge rubric are also English throughout, while the claims span 41 languages.

6.7. Future Work

The clearest opening is to give the pipeline the explicit sourcing stage that Section 6.3 finds missing, so that something in the design decides which sources a claim deserves. Material for a first attempt already exists, since the article analysis of Section 4.4 records which sources settled each training claim, and a quarter of them resolve to a site directly. Learning a distribution over source classes conditioned on claim type, then issuing site-scoped retrieval against it, is the version of this that fits the present architecture. The blueprint-shaped version extends the representation so that a procedure names where to look alongside what to establish, which is the field it currently lacks. Both are testable against gold-source recall without running the pipeline end to end.

The second change follows from Section 6.2, and the aim is a hedge that reflects the evidence rather than the state of the checklist. Uncertainty is the right output when the evidence is thin, so the problem is what currently produces it. An open check pulls the verdict toward the middle whether it concerns the decisive question or a branch the claim never needed, while a check closed on a single weak page counts the same

as one closed on a decisive source. Certainty should instead follow the strength and consistency of what was gathered, with an unfinished procedure entering only where the unresolved check is the one that would settle the claim. The population this affects is already identified, namely the claims where the blueprint hedges and the baseline commits, whose error of 0.731 against 0.357 bounds what the change is worth. Part of that gap reflects a benchmark whose ground truth almost never lands in the neutral band, so the target is a better-founded hedge rather than a rarer one.

Third, conditioning could operate on individual checks rather than whole procedures. Claim 103074 (Figure 5.6) fails because both of its active checks are satisfiable without touching what is false about the claim, which is a composition problem rather than a selection problem. A system that assembles the checks a claim needs, instead of choosing one graph that carries them, would also sidestep the consolidation that compressed fourteen clusters into four blueprints.

Fourth, the judge boundary deserves a relevance stage that can run at inference time. Surplus evidence costs real accuracy even when nothing is missing (Section 5.3.2), and the one deployable filter tested here hurt on a second run. The judge boundary needs a filter trained to keep what a judge will actually use, rather than one that drops what is off-topic.

7. Conclusion

This thesis asked whether an automated fact-checker benefits from choosing its verification procedure per claim. It learned a pool of verification blueprints from published fact-checks and conditioned the pipeline on one per claim. Verification procedures can be recovered from the published record, and the learning needs no annotated traces. Claim text alone predicts which procedure a fact-checker followed at 83.9% on a held-out quarter, above the majority baseline, and a linear probe over claim embeddings does this better than an LLM asked to choose.

Conditioning on the recovered procedure does not improve accuracy. It helped the weaker of two planners and hurt the stronger, with all five judges agreeing on both directions. The routed pool never reached what the stronger planner achieves unguided. Guidance of this form substitutes for a planner’s own procedure rather than adding to it, so its value falls as planners improve. One property of the design accounts for both halves. A blueprint’s checks occupy separate cells, so an unsettled cell keeps the search running where the missing evidence is reachable, and drags the verdict toward the middle of the scale otherwise.

The error that remains comes from retrieval, and within retrieval from the choice of which sources to consult. Judging on the evidence a human fact-checker gathered cuts error by 85 to 88% and leaves under one percent of claims judged in the wrong direction. A 35B judge is indistinguishable from a frontier model on that task. Retrieval recovers under half of those findings, and pipeline error tracks that recall while claim difficulty does not. Four in five of the sources the fact-checkers used were never put in front of the retriever. The stage that produces the verdict is close to saturated. The stage that decides which sources to consult was never built.

Three component results stand apart from that argument and hold independently of it. General-purpose multimodal models beat specialized geolocation models by a wide margin on claim imagery, on public and never-published photographs alike. Manipulation forensics does not survive redistribution, and even a perfect detector leaves the verdict where it was. Verdict certainty is not recoverable from execution-trace features.

What follows for this design is that effort belongs upstream of everything the strategy layer controls. Procedural knowledge learned from published fact-checks would have to carry where to look alongside what to establish before it could act on the constraint, and the record it is learned from already names the sources that settled each case. The diagnosis is also cheap to repeat. Recall against those sources can be computed for any system built on a fact-check-derived benchmark, with no new annotation, and reporting it beside an end-to-end score separates a system that reasoned badly from one that never saw the page. Establishing that distinction is what this thesis spent its evidence on.



Acknowledgements

We gratefully acknowledge support from the hessian.AI Service Center (funded by the Federal Ministry of Research, Technology and Space, BMFTR, grant no. 16IS22091) and the hessian.AI Innovation Lab (funded by the Hessian Ministry for Digital Strategy and Innovation, grant no. S-DIW04/0013/003).

Bibliography

- [1] Sahar Abdelnabi, Rakibul Hasan, and Mario Fritz. “Open-Domain, Content-based, Multi-modal Fact-checking of Out-of-Context Images via Online Resources”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). ISSN: 2575-7075. June 2022, pp. 14920–14929. DOI: 10.1109/CVPR52688.2022.01452. URL: <https://ieeexplore.ieee.org/document/9878739> (visited on 08/24/2026).
- [2] Raia Abu Ahmad, Aida Usmanova, and Georg Rehm. “The ClimateCheck Shared Task: Scientific Fact-Checking of Social Media Claims about Climate Change”. In: *Proceedings of the Fifth Workshop on Scholarly Document Processing (SDP 2025)*. Ed. by Tirthankar Ghosal et al. Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 263–275. ISBN: 979-8-89176-265-7. DOI: 10.18653/v1/2025.sdp-1.24. URL: <https://aclanthology.org/2025.sdp-1.24/> (visited on 07/01/2026).
- [3] Chirag Agarwal, Sree Harsha Tanneru, and Himabindu Lakkaraju. *Faithfulness vs. Plausibility: On the (Un)Reliability of Explanations from Large Language Models*. Mar. 14, 2024. DOI: 10.48550/arXiv.2402.04614. arXiv: 2402.04614[cs]. URL: <http://arxiv.org/abs/2402.04614> (visited on 09/21/2025).
- [4] Mubashara Akhtar et al. “Multimodal Automated Fact-Checking: A Survey”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Findings 2023. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 5430–5448. DOI: 10.18653/v1/2023.findings-emnlp.361. URL: <https://aclanthology.org/2023.findings-emnlp.361/> (visited on 06/29/2026).
- [5] Anirban Saha Anik et al. “Multi-Agent Retrieval-Augmented Framework for Evidence-Based Counter-speech Against Health Misinformation”. In: *Second Conference on Language Modeling*. Aug. 26, 2025. URL: <https://openreview.net/forum?id=P61AgRyU7E#discussion> (visited on 08/24/2026).
- [6] Anthropic. *Claude Opus 4.6*. 2026. URL: <https://www.anthropic.com/news/claude-opus-4-6>.
- [7] Anthropic. *Claude Opus 4.8*. 2026. URL: <https://www.anthropic.com/news/claude-opus-4-8>.
- [8] Anthropic. *Claude Opus 5*. 2026. URL: <https://www.anthropic.com/news/claude-opus-5>.
- [9] Iván Arcuschin et al. “Chain-of-Thought Reasoning in the Wild is not Always Faithful”. In: *Workshop on Reasoning and Planning for Large Language Models*. Mar. 5, 2025. URL: <https://openreview.net/forum?id=L8094Whth0> (visited on 08/24/2026).
- [10] Pepa Atanasova et al. “Generating Fact Checking Explanations”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, July 2020, pp. 7352–7364. DOI: 10.18653/v1/2020.acl-main.656. URL: <https://aclanthology.org/2020.acl-main.656/> (visited on 07/23/2026).
- [11] Bowen Baker et al. *Monitoring Reasoning Models for Misbehavior and the Risks of Promoting Obfuscation*. Mar. 14, 2025. DOI: 10.48550/arXiv.2503.11926. arXiv: 2503.11926[cs.AI]. URL: <http://arxiv.org/abs/2503.11926> (visited on 07/06/2026).

-
- [12] Dibyanayan Bandyopadhyay, Soham Bhattacharjee, and Asif Ekbal. *Thinking Machines: A Survey of LLM based Reasoning Strategies*. Mar. 13, 2025. DOI: 10.48550/arXiv.2503.10814. arXiv: 2503.10814[cs]. URL: <http://arxiv.org/abs/2503.10814> (visited on 09/21/2025).
- [13] Xiaohe Bo et al. “Reflective Multi-Agent Collaboration based on Large Language Models”. In: *Advances in Neural Information Processing Systems* 37 (Dec. 16, 2024), pp. 138595–138631. URL: https://proceedings.neurips.cc/paper_files/paper/2024/hash/fa54b0edce5eef0bb07654e8ee800cb4-Abstract-Conference.html (visited on 10/07/2025).
- [14] Matthew Bowers et al. “Top-Down Synthesis for Library Learning”. In: *Proceedings of the ACM on Programming Languages* 7 (POPL Jan. 9, 2023), pp. 1182–1213. ISSN: 2475-1421. DOI: 10.1145/3571234. arXiv: 2211.16605[cs.PL]. URL: <http://arxiv.org/abs/2211.16605> (visited on 07/21/2026).
- [15] Samuel R. Bowman et al. “A large annotated corpus for learning natural language inference”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2015. Ed. by Lluís Màrquez, Chris Callison-Burch, and Jian Su. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 632–642. DOI: 10.18653/v1/D15-1075. URL: <https://aclanthology.org/D15-1075/> (visited on 08/24/2026).
- [16] Tobias Braun et al. “DEFAME: Dynamic Evidence-based FACT-checking with Multimodal Experts”. In: *Proceedings of the 42nd International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, Oct. 6, 2025, pp. 5383–5417. URL: <https://proceedings.mlr.press/v267/braun25b.html> (visited on 08/24/2026).
- [17] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. “Density-Based Clustering Based on Hierarchical Density Estimates”. In: *Advances in Knowledge Discovery and Data Mining*. Ed. by Jian Pei et al. Berlin, Heidelberg: Springer, 2013, pp. 160–172. ISBN: 978-3-642-37456-2. DOI: 10.1007/978-3-642-37456-2_14.
- [18] Rui Cao et al. *AverImaTeC: A Dataset for Automatic Verification of Image-Text Claims with Evidence from the Web*. Oct. 7, 2025. DOI: 10.48550/arXiv.2505.17978. arXiv: 2505.17978[cs]. URL: <http://arxiv.org/abs/2505.17978> (visited on 12/23/2025).
- [19] Mert Cemri et al. “Why Do Multi-Agent LLM Systems Fail?” In: *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. Oct. 30, 2025. URL: <https://openreview.net/forum?id=fAjbYBmonr> (visited on 08/27/2026).
- [20] Vicente Vivanco Cepeda, Gaurav Kumar Nayak, and Mubarak Shah. “GeoCLIP: Clip-inspired alignment between locations and images for effective worldwide geo-localization”. In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS ’23. Red Hook, NY, USA: Curran Associates Inc., Dec. 10, 2023, pp. 8690–8701. (Visited on 08/24/2026).
- [21] Canyu Chen and Kai Shu. “Can LLM-Generated Misinformation Be Detected?” In: *The Twelfth International Conference on Learning Representations*. Oct. 13, 2023. URL: <https://openreview.net/forum?id=ccxD4mtkTU> (visited on 08/24/2026).
- [22] Wenhui Chen et al. *Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks*. Oct. 23, 2023. DOI: 10.48550/arXiv.2211.12588. arXiv: 2211.12588[cs.CL]. URL: <http://arxiv.org/abs/2211.12588> (visited on 07/07/2026).
- [23] Xingyu Chen et al. “Do NOT Think That Much for $2+3=?$ On the Overthinking of Long Reasoning Models”. In: *Forty-second International Conference on Machine Learning*. June 18, 2025. URL: <https://openreview.net/forum?id=MSbU3L7V00> (visited on 08/27/2026).

-
- [24] Yanda Chen et al. *Reasoning Models Don't Always Say What They Think*. May 8, 2025. DOI: 10.48550/arXiv.2505.05410. arXiv: 2505.05410[cs.CL]. URL: <http://arxiv.org/abs/2505.05410> (visited on 07/06/2026).
- [25] Zacharias Chrysidis et al. “Credible, Unreliable or Leaked?: Evidence verification for enhanced automated fact-checking”. In: *Proceedings of the 3rd ACM International Workshop on Multimedia AI against Disinformation*. MAD ’24. New York, NY, USA: Association for Computing Machinery, June 10, 2024, pp. 73–81. ISBN: 979-8-4007-0552-6. DOI: 10.1145/3643491.3660278. URL: <https://dl.acm.org/doi/10.1145/3643491.3660278> (visited on 07/01/2026).
- [26] James Chua and Owain Evans. “Are DeepSeek R1 And Other Reasoning Models More Faithful?” In: *ICLR 2025 Workshop on Foundation Models in the Wild*. Mar. 6, 2025. URL: <https://openreview.net/forum?id=rI38nonvF5> (visited on 08/27/2026).
- [27] Yuwei Chuai et al. “Community-based fact-checking reduces the spread of misleading posts on X (formerly Twitter)”. In: *Nature Communications* 17.1 (May 5, 2026), p. 4070. ISSN: 2041-1723. DOI: 10.1038/s41467-026-72597-0. URL: <https://www.nature.com/articles/s41467-026-72597-0> (visited on 06/02/2026).
- [28] Karl Cobbe et al. *Training Verifiers to Solve Math Word Problems*. Nov. 18, 2021. DOI: 10.48550/arXiv.2110.14168. arXiv: 2110.14168[cs.LG]. URL: <http://arxiv.org/abs/2110.14168> (visited on 07/02/2026).
- [29] Alejandro Cuadron et al. *The Danger of Overthinking: Examining the Reasoning-Action Dilemma in Agentic Tasks*. Feb. 12, 2025. DOI: 10.48550/arXiv.2502.08235. arXiv: 2502.08235[cs.AI]. URL: <http://arxiv.org/abs/2502.08235> (visited on 07/06/2026).
- [30] Xing Cui et al. “T2Agent: A Tool-augmented Multimodal Misinformation Detection Agent with Monte Carlo Tree Search”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 40.1 (Mar. 14, 2026), pp. 175–183. ISSN: 2374-3468. DOI: 10.1609/aaai.v40i1.36977. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/36977> (visited on 08/24/2026).
- [31] Ido Dagan, Oren Glickman, and Bernardo Magnini. “The PASCAL Recognising Textual Entailment Challenge”. In: *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*. Ed. by Joaquin Quiñonero-Candela et al. Berlin, Heidelberg: Springer, 2006, pp. 177–190. ISBN: 978-3-540-33428-6. DOI: 10.1007/11736790_9.
- [32] DeepSeek-AI et al. “DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning”. In: *Nature* 645.8081 (Sept. 18, 2025), pp. 633–638. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/s41586-025-09422-z. arXiv: 2501.12948[cs.CL]. URL: <http://arxiv.org/abs/2501.12948> (visited on 07/02/2026).
- [33] Yilun Du et al. “Improving factuality and reasoning in language models through multiagent debate”. In: *Proceedings of the 41st International Conference on Machine Learning*. Vol. 235. ICML’24. Vienna, Austria: JMLR.org, July 21, 2024, pp. 11733–11763. (Visited on 08/26/2026).
- [34] Nicholas Dufour et al. “Ammeba: A large-scale survey and dataset of media-based misinformation in-the-wild”. In: ().
- [35] Kevin Ellis et al. *DreamCoder: Growing generalizable, interpretable knowledge with wake-sleep Bayesian program learning*. June 15, 2020. DOI: 10.48550/arXiv.2006.08381. arXiv: 2006.08381[cs.AI]. URL: <http://arxiv.org/abs/2006.08381> (visited on 07/21/2026).

-
- [36] Jiazhan Feng et al. “ReTool: Reinforcement Learning for Strategic Tool Use in LLMs”. In: *International Conference on Learning Representations 2026* (Apr. 20, 2026), pp. 37909–37926. URL: https://proceedings.iclr.cc/paper_files/paper/2026/hash/4038c9208dfc22644c60ad39c24e5c53-Abstract-Conference.html (visited on 08/27/2026).
- [37] Luyu Gao et al. “PAL: program-aided language models”. In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. ICML’23. Honolulu, Hawaii, USA: JMLR.org, July 23, 2023, pp. 10764–10799. (Visited on 08/27/2026).
- [38] Jiahui Geng, Jonathan Tonglet, and Iryna Gurevych. *M4FC: a Multimodal, Multilingual, Multicultural, Multitask Real-World Fact-Checking Dataset*. June 10, 2026. DOI: 10.48550/arXiv.2510.23508. arXiv: 2510.23508[cs.CL]. URL: <http://arxiv.org/abs/2510.23508> (visited on 07/08/2026).
- [39] Google. *Gemini 3 Flash*. 2026. URL: <https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/gemini/3-flash>.
- [40] Google. *Gemini 3.5 Flash*. 2026. URL: <https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/gemini/3-5-flash>.
- [41] Zhibin Gou et al. “CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing”. In: *The Twelfth International Conference on Learning Representations*. Oct. 13, 2023. URL: <https://openreview.net/forum?id=Sx038qxjek> (visited on 08/26/2026).
- [42] Gabriel Grand et al. “LILO: Learning Interpretable Libraries by Compressing and Documenting Code”. In: *The Twelfth International Conference on Learning Representations*. Oct. 13, 2023. URL: <https://openreview.net/forum?id=TqYbAWKMIe> (visited on 08/24/2026).
- [43] Lucas Graves. “Anatomy of a Fact Check: Objective Practice and the Contested Epistemology of Fact Checking”. In: *Communication, Culture and Critique* 10.3 (Sept. 1, 2017), pp. 518–537. ISSN: 1753-9129. DOI: 10.1111/cccr.12163. URL: <https://doi.org/10.1111/cccr.12163> (visited on 07/23/2026).
- [44] Kai Greshake et al. “Not What You’ve Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection”. In: *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*. AISec ’23. New York, NY, USA: Association for Computing Machinery, Nov. 26, 2023, pp. 79–90. ISBN: 979-8-4007-0260-0. DOI: 10.1145/3605764.3623985. URL: <https://dl.acm.org/doi/10.1145/3605764.3623985> (visited on 08/27/2026).
- [45] Maarten Grootendorst. *BERTopic: Neural topic modeling with a class-based TF-IDF procedure*. Mar. 11, 2022. DOI: 10.48550/arXiv.2203.05794. arXiv: 2203.05794[cs.CL]. URL: <http://arxiv.org/abs/2203.05794> (visited on 07/21/2026).
- [46] Yimeng Gu et al. *Multi-MLLM Knowledge Distillation for Out-of-Context News Detection*. May 28, 2025. DOI: 10.48550/arXiv.2505.22517. arXiv: 2505.22517[cs.CL]. URL: <http://arxiv.org/abs/2505.22517> (visited on 06/30/2026).
- [47] Fabrizio Guillaro et al. “TruFor: Leveraging All-Round Clues for Trustworthy Image Forgery Detection and Localization”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 20606–20615.
- [48] Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. “A Survey on Automated Fact-Checking”. In: *Transactions of the Association for Computational Linguistics* 10 (2022). Ed. by Brian Roark and Ani Nenkova, pp. 178–206. DOI: 10.1162/tacl_a_00454. URL: <https://aclanthology.org/2022.tacl-1.11/> (visited on 06/28/2026).
- [49] Lukas Haas, Silas Alberti, and Michal Skreta. *Learning Generalized Zero-Shot Learners for Open-Domain Image Geolocalization*. 2023. arXiv: 2302.00275 [cs.CV].

-
- [50] Shanshan Han et al. *LLM Multi-Agent Systems: Challenges and Open Problems*. May 12, 2025. DOI: 10.48550/arXiv.2402.03578. arXiv: 2402.03578[cs]. URL: <http://arxiv.org/abs/2402.03578> (visited on 10/01/2025).
- [51] Ching Nam Hang, Pei-Duo Yu, and Chee Wei Tan. “TrumorGPT: Graph-Based Retrieval-Augmented Large Language Model for Fact-Checking”. In: *IEEE Transactions on Artificial Intelligence* 6.11 (Nov. 2025), pp. 3148–3162. ISSN: 2691-4581. DOI: 10.1109/TAI.2025.3567369. URL: <https://ieeexplore.ieee.org/abstract/document/10988740> (visited on 07/01/2026).
- [52] Shibo Hao et al. “Training Large Language Models to Reason in a Continuous Latent Space”. In: *Second Conference on Language Modeling*. Aug. 26, 2025. URL: <https://openreview.net/forum?id=Itxz7S4Ip3#discussion> (visited on 08/24/2026).
- [53] Yuren Hao, Xiang Wan, and ChengXiang Zhai. *An Investigation of Robustness of LLMs in Mathematical Reasoning: Benchmarking with Mathematically-Equivalent Transformation of Advanced Mathematical Problems*. Dec. 4, 2025. DOI: 10.48550/arXiv.2508.08833. arXiv: 2508.08833[cs.CL]. URL: <http://arxiv.org/abs/2508.08833> (visited on 07/06/2026).
- [54] Haorui He et al. “Debating Truth: Debate-driven Claim Verification with Multiple Large Language Model Agents”. In: *Proceedings of the ACM Web Conference 2026*. Apr. 13, 2026, pp. 8851–8861. DOI: 10.1145/3774904.3792993. arXiv: 2507.19090[cs.CL]. URL: <http://arxiv.org/abs/2507.19090> (visited on 06/22/2026).
- [55] Lukas Helff et al. *LLMs Gaming Verifiers: RLVR can Lead to Reward Hacking*. Apr. 16, 2026. DOI: 10.48550/arXiv.2604.15149. arXiv: 2604.15149[cs.LG]. URL: <http://arxiv.org/abs/2604.15149> (visited on 07/06/2026).
- [56] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. Mar. 9, 2015. DOI: 10.48550/arXiv.1503.02531. arXiv: 1503.02531[stat]. URL: <http://arxiv.org/abs/1503.02531> (visited on 09/17/2025).
- [57] Tran Thai Hoa et al. *ViFactCheck: A New Benchmark Dataset and Methods for Multi-domain News Fact-Checking in Vietnamese*. Dec. 19, 2024. DOI: 10.48550/arXiv.2412.15308. arXiv: 2412.15308[cs.CL]. URL: <http://arxiv.org/abs/2412.15308> (visited on 07/21/2026).
- [58] Or Honovich et al. “Instruction Induction: From Few Examples to Natural Language Task Descriptions”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2023. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 1935–1952. DOI: 10.18653/v1/2023.acl-long.108. URL: <https://aclanthology.org/2023.acl-long.108/> (visited on 07/21/2026).
- [59] Yutao Hou et al. “Toward Automated Robustness Evaluation of Mathematical Reasoning”. In: *Findings of the Association for Computational Linguistics: ACL 2026*. Findings 2026. Ed. by Maria Liakata et al. San Diego, California, United States: Association for Computational Linguistics, July 2026, pp. 14117–14139. ISBN: 979-8-89176-395-1. DOI: 10.18653/v1/2026.findings-acl.691. URL: <https://aclanthology.org/2026.findings-acl.691/> (visited on 08/27/2026).
- [60] Cheng-Yu Hsieh et al. “Distilling Step-by-Step! Outperforming Larger Language Models with Less Training Data and Smaller Model Sizes”. In: *Findings of the Association for Computational Linguistics: ACL 2023*. Findings 2023. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 8003–8017. DOI: 10.18653/v1/2023.findings-acl.507. URL: <https://aclanthology.org/2023.findings-acl.507/> (visited on 08/27/2026).

-
- [61] Jingcheng Hu et al. “Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model”. In: *Advances in Neural Information Processing Systems*. Vol. 38, Main Conference. Curran Associates, Inc., 2025, pp. 162239–162262. DOI: 10.52202/085713–5418. URL: https://proceedings.neurips.cc/paper_files/paper/2025/hash/ed873d79e7c268c020c4b4db13a2812Abstract-Conference.html (visited on 08/27/2026).
- [62] Shengran Hu, Cong Lu, and Jeff Clune. “Automated Design of Agentic Systems”. In: *The Thirteenth International Conference on Learning Representations*. Oct. 4, 2024. URL: <https://openreview.net/forum?id=t9U3LW7JVX> (visited on 08/24/2026).
- [63] Yue Hu et al. “Self-Evolving Multi-Agent Collaboration Networks for Software Development”. In: *The Thirteenth International Conference on Learning Representations*. Oct. 4, 2024. URL: <https://openreview.net/forum?id=4R71pdPBZp> (visited on 08/27/2026).
- [64] Jie Huang and Kevin Chen-Chuan Chang. “Towards Reasoning in Large Language Models: A Survey”. In: *Findings of the Association for Computational Linguistics: ACL 2023*. Findings 2023. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 1049–1065. DOI: 10.18653/v1/2023.findings-acl.67. URL: <https://aclanthology.org/2023.findings-acl.67/> (visited on 08/24/2026).
- [65] P. N. Johnson-Laird. “Mental models in cognitive science”. In: *Cognitive Science* 4.1 (Jan. 1, 1980), pp. 71–115. ISSN: 0364-0213. DOI: 10.1016/S0364-0213(81)80005-5. URL: <https://www.sciencedirect.com/science/article/pii/S0364021381800055> (visited on 09/21/2025).
- [66] Jaeyoon Jung, Yejun Yoon, and Kunwoo Park. *Is a Picture Worth a Thousand Words? Adaptive Multimodal Fact-Checking with Visual Evidence Necessity*. May 13, 2026. DOI: 10.48550/arXiv.2604.04692. arXiv: 2604.04692[cs.CL]. URL: <http://arxiv.org/abs/2604.04692> (visited on 06/22/2026).
- [67] Ehud Karpas et al. *MRKL Systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning*. May 1, 2022. DOI: 10.48550/arXiv.2205.00445. arXiv: 2205.00445[cs.CL]. URL: <http://arxiv.org/abs/2205.00445> (visited on 07/07/2026).
- [68] Takeshi Kojima et al. “Large Language Models are Zero-Shot Reasoners”. In: *Advances in Neural Information Processing Systems*. Vol. 35. Curran Associates, Inc., 2022, pp. 22199–22213. DOI: 10.52202/068431–1613. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html (visited on 08/27/2026).
- [69] Lea Krause et al. “Confidently Wrong: Exploring the Calibration and Expression of (Un)Certainty of Large Language Models in a Multilingual Setting”. In: *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*. Ed. by Albert Gatt et al. Prague, Czech Republic: Association for Computational Linguistics, Sept. 2023, pp. 1–9. URL: <https://aclanthology.org/2023.mmnlg-1.1/> (visited on 08/13/2026).
- [70] Tamera Lanham et al. *Measuring Faithfulness in Chain-of-Thought Reasoning*. July 17, 2023. DOI: 10.48550/arXiv.2307.13702. arXiv: 2307.13702[cs.AI]. URL: <http://arxiv.org/abs/2307.13702> (visited on 07/06/2026).
- [71] Markus Leippold et al. “Automated fact-checking of climate claims with large language models”. In: *npj Climate Action* 4.1 (Feb. 25, 2025), p. 17. ISSN: 2731-9814. DOI: 10.1038/s44168-025-00215-8. URL: <https://www.nature.com/articles/s44168-025-00215-8> (visited on 06/30/2026).

-
- [72] Fanxiao Li et al. “CMIE: Combining MLLM Insights with External Evidence for Explainable Out-of-Context Misinformation Detection”. In: *Findings of the Association for Computational Linguistics: ACL 2025*. Findings 2025. Ed. by Wanxiang Che et al. Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 9342–9354. ISBN: 979-8-89176-256-5. DOI: 10.18653/v1/2025.findings-acl.487. URL: <https://aclanthology.org/2025.findings-acl.487/> (visited on 06/30/2026).
- [73] Haiwen Li and Michiel A. Bakker. *AI Fact-Checking in the Wild: A Field Evaluation of LLM-Written Community Notes on X*. Version Number: 2. 2026. DOI: 10.48550/ARXIV.2604.02592. URL: <https://arxiv.org/abs/2604.02592> (visited on 06/02/2026).
- [74] Haonan Li et al. “Loki: An Open-Source Tool for Fact Verification”. In: *Proceedings of the 31st International Conference on Computational Linguistics: System Demonstrations*. Ed. by Owen Rambow et al. Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 28–36. URL: <https://aclanthology.org/2025.coling-demos.4/> (visited on 12/20/2025).
- [75] Minghao Li et al. “API-Bank: A Comprehensive Benchmark for Tool-Augmented LLMs”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2023. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 3102–3116. DOI: 10.18653/v1/2023.emnlp-main.187. URL: <https://aclanthology.org/2023.emnlp-main.187/> (visited on 08/26/2026).
- [76] Siheng Li et al. “A Survey on the Honesty of Large Language Models”. In: *Transactions on Machine Learning Research* (Oct. 30, 2024). ISSN: 2835-8856. URL: <https://openreview.net/forum?id=FJgtVfUxLQ> (visited on 08/27/2026).
- [77] Tianxiao Li et al. *Omni-Fake: Benchmarking Unified Multimodal Social Media Deepfake Detection*. May 2, 2026. DOI: 10.48550/arXiv.2605.01638. arXiv: 2605.01638[cs.CV]. URL: <http://arxiv.org/abs/2605.01638> (visited on 07/21/2026).
- [78] Xinyi Li et al. “A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges”. In: *Vicinagearth 1.1* (Oct. 8, 2024), p. 9. ISSN: 3005-060X. DOI: 10.1007/s44336-024-00009-2. URL: <https://doi.org/10.1007/s44336-024-00009-2> (visited on 10/01/2025).
- [79] Xuechen Liang et al. “Self-evolving Agents with reflective and memory-augmented abilities”. In: *LLM-based Multi-Agent Systems: Towards Responsible, Reliable, and Scalable Agentic Systems*. Jan. 10, 2026. URL: <https://openreview.net/forum?id=6Mw2f03ejN> (visited on 08/27/2026).
- [80] Fanzhen Liu et al. “Adversarial Attacks Against Automated Fact-Checking: A Survey”. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2025. Ed. by Christos Christodoulopoulos et al. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 22968–22990. ISBN: 979-8-89176-332-6. DOI: 10.18653/v1/2025.emnlp-main.1171. URL: <https://aclanthology.org/2025.emnlp-main.1171/> (visited on 07/08/2026).
- [81] Xuannan Liu et al. *MMFakeBench: A Mixed-Source Multimodal Misinformation Detection Benchmark for LVLMS*. Feb. 27, 2025. DOI: 10.48550/arXiv.2406.08772. arXiv: 2406.08772[cs.CV]. URL: <http://arxiv.org/abs/2406.08772> (visited on 07/21/2026).
- [82] Grace Luo, Trevor Darrell, and Anna Rohrbach. “NewsCLIPPings: Automatic Generation of Out-of-Context Multimodal Media”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2021. Ed. by Marie-Francine Moens et al. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6801–6817. DOI: 10.18653/v1/2021.emnlp-main.545. URL: <https://aclanthology.org/2021.emnlp-main.545/> (visited on 07/21/2026).

-
- [83] Jiatong Ma et al. “LoCal: Logical and Causal Fact-Checking with LLM-Based Multi-Agents”. In: *Proceedings of the ACM on Web Conference 2025*. WWW ’25. New York, NY, USA: Association for Computing Machinery, Apr. 22, 2025, pp. 1614–1625. ISBN: 979-8-4007-1274-6. DOI: 10.1145/3696410.3714748. URL: <https://dl.acm.org/doi/10.1145/3696410.3714748> (visited on 12/17/2025).
- [84] Xiaowen Ma et al. “Self-Evolving Multi-Agent Systems via Textual Backpropagation”. In: *Findings of the Association for Computational Linguistics: ACL 2026*. Findings 2026. Ed. by Maria Liakata et al. San Diego, California, United States: Association for Computational Linguistics, July 2026, pp. 9918–9951. ISBN: 979-8-89176-395-1. DOI: 10.18653/v1/2026.findings-acl.483. URL: <https://aclanthology.org/2026.findings-acl.483/> (visited on 08/26/2026).
- [85] Aman Madaan et al. “Self-Refine: Iterative Refinement with Self-Feedback”. In: *Advances in Neural Information Processing Systems*. Vol. 36. Curran Associates, Inc., 2023, pp. 46534–46594. DOI: 10.52202/075280–2019. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html (visited on 08/27/2026).
- [86] Lucie Charlotte Magister et al. “Teaching Small Language Models to Reason”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. ACL 2023. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 1773–1781. DOI: 10.18653/v1/2023.acl-short.151. URL: <https://aclanthology.org/2023.acl-short.151/> (visited on 08/27/2026).
- [87] Leland McInnes, John Healy, and Steve Astels. “hdbscan: Hierarchical density based clustering”. In: *Journal of Open Source Software* 2.11 (Mar. 21, 2017), p. 205. ISSN: 2475-9066. DOI: 10.21105/joss.00205. URL: <https://joss.theoj.org/papers/10.21105/joss.00205> (visited on 08/16/2026).
- [88] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. Sept. 18, 2020. DOI: 10.48550/arXiv.1802.03426. arXiv: 1802.03426[stat.ML]. URL: <http://arxiv.org/abs/1802.03426> (visited on 08/03/2026).
- [89] Grégoire Mialon et al. “Augmented Language Models: a Survey”. In: *Transactions on Machine Learning Research* (Mar. 20, 2023). ISSN: 2835-8856. URL: <https://openreview.net/forum?id=jh7wH2AzKK> (visited on 08/26/2026).
- [90] Nicholas Micallef et al. “True or False: Studying the Work Practices of Professional Fact-Checkers”. In: *Proceedings of the ACM on Human-Computer Interaction* 6 (CSCW1 Apr. 7, 2022), 127:1–127:44. DOI: 10.1145/3512974. URL: <https://dl.acm.org/doi/10.1145/3512974> (visited on 07/23/2026).
- [91] Seyed Iman Mirzadeh et al. “GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models”. In: *The Thirteenth International Conference on Learning Representations*. Oct. 4, 2024. URL: <https://openreview.net/forum?id=AjXkRZIVjB> (visited on 08/27/2026).
- [92] Guangyu Mu et al. “CLAAF: Multimodal fake information detection based on contrastive learning and adaptive Agg-modality fusion”. In: *PLOS ONE* 20.5 (May 7, 2025), e0322556. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0322556. URL: <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0322556> (visited on 12/10/2025).
- [93] P. Nakov et al. “Overview of the CLEF-2022 CheckThat! Lab Task 2 on Detecting Previously Fact-Checked Claims”. In: *Experimental IR Meets Multilinguality, Multimodality, and Interaction*. Accepted: 2024-10-22T04:30:25Z. CEUR-WS, 2022. URL: <https://www.research.unipd.it/handle/11577/3537005> (visited on 07/23/2026).
- [94] Kei Nishimura-Gasparian, Robert McCarthy, and David Lindner. *Towards Understanding Specification Gaming in Reasoning Models*. May 4, 2026. DOI: 10.48550/arXiv.2605.02269. arXiv: 2605.02269[cs.AI]. URL: <http://arxiv.org/abs/2605.02269> (visited on 07/06/2026).

-
- [95] Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Advances in Neural Information Processing Systems*. Vol. 35. Curran Associates, Inc., 2022, pp. 27730–27744. DOI: 10.52202/068431–2011. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract.html (visited on 08/27/2026).
- [96] Charles Packer et al. *MemGPT: Towards LLMs as Operating Systems*. Feb. 12, 2024. DOI: 10.48550/arXiv.2310.08560. arXiv: 2310.08560[cs.AI]. URL: <http://arxiv.org/abs/2310.08560> (visited on 07/08/2026).
- [97] Stefanos-Iordanis Papadopoulos et al. “RED-DOT: Multimodal Fact-Checking via Relevant Evidence Detection”. In: *IEEE Transactions on Computational Social Systems* 12.6 (Dec. 2025), pp. 4231–4240. ISSN: 2329-924X. DOI: 10.1109/TCSS.2025.3553939. URL: <https://ieeexplore.ieee.org/abstract/document/10948326> (visited on 12/16/2025).
- [98] Stefanos-Iordanis Papadopoulos et al. *VERITE: A Robust Benchmark for Multimodal Misinformation Detection Accounting for Unimodal Bias*. Jan. 8, 2024. DOI: 10.48550/arXiv.2304.14133. arXiv: 2304.14133[cs]. URL: <http://arxiv.org/abs/2304.14133> (visited on 12/09/2025).
- [99] Shishir G. Patil et al. “Gorilla: Large Language Model Connected with Massive APIs”. In: *Advances in Neural Information Processing Systems*. Vol. 37. Curran Associates, Inc., 2024, pp. 126544–126565. DOI: 10.52202/079017–4020. URL: https://proceedings.neurips.cc/paper_files/paper/2024/hash/e4c61f578ff07830f5c37378dd3ecb0d-Abstract-Conference.html (visited on 08/26/2026).
- [100] Reid Pryzant et al. “Automatic Prompt Optimization with “Gradient Descent” and Beam Search”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2023. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 7957–7968. DOI: 10.18653/v1/2023.emnlp-main.494. URL: <https://aclanthology.org/2023.emnlp-main.494/> (visited on 07/21/2026).
- [101] Akshith Reddy Putta, Jacob Devasier, and Chengkai Li. “ClaimCheck: Automatic Fact-Checking of Textual Claims using Web Evidence”. In: *Proceedings of the 4th International Workshop on Knowledge-Augmented Methods for Natural Language Processing*. Ed. by Weijia Shi et al. Albuquerque, New Mexico, USA: Association for Computational Linguistics, May 2025, pp. 303–316. ISBN: 979-8-89176-229-9. DOI: 10.18653/v1/2025.knowledgenlp-1.26. URL: <https://aclanthology.org/2025.knowledgenlp-1.26/> (visited on 06/22/2026).
- [102] Peng Qi et al. *FakeSV: A Multimodal Benchmark with Rich Social Context for Fake News Detection on Short Video Platforms*. Dec. 2, 2022. DOI: 10.48550/arXiv.2211.10973. arXiv: 2211.10973[cs.MM]. URL: <http://arxiv.org/abs/2211.10973> (visited on 07/21/2026).
- [103] Peng Qi et al. “Sniffer: Multimodal Large Language Model for Explainable Out-of-Context Misinformation Detection”. In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). ISSN: 2575-7075. June 2024, pp. 13052–13062. DOI: 10.1109/CVPR52733.2024.01240. URL: <https://ieeexplore.ieee.org/document/10656149> (visited on 08/24/2026).
- [104] Cheng Qian et al. “CREATOR: Tool Creation for Disentangling Abstract and Concrete Reasoning of Large Language Models”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Findings 2023. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 6922–6939. DOI: 10.18653/v1/2023.findings-emnlp.462. URL: <https://aclanthology.org/2023.findings-emnlp.462/> (visited on 07/21/2026).

-
- [105] Yujia Qin et al. “ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs”. In: *The Twelfth International Conference on Learning Representations*. Oct. 13, 2023. URL: <https://openreview.net/forum?id=dHng200Jjr> (visited on 08/27/2026).
- [106] Qwen Team. *Qwen3.5: Towards Native Multimodal Agents*. Feb. 2026. URL: <https://qwen.ai/blog?id=qwen3.5>.
- [107] Qwen Team. *Qwen3.6-35B-A3B: Agentic Coding Power, Now Open to All*. Apr. 2026. URL: <https://qwen.ai/blog?id=qwen3.6-35b-a3b>.
- [108] Rafael Rafailov et al. “Direct Preference Optimization: Your Language Model is Secretly a Reward Model”. In: *Advances in Neural Information Processing Systems*. Vol. 36. Curran Associates, Inc., 2023, pp. 53728–53741. DOI: 10.52202/075280–2338. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html (visited on 08/27/2026).
- [109] Aman Rangapur et al. *Fin-Fact: A Benchmark Dataset for Multimodal Financial Fact Checking and Explanation Generation*. May 1, 2024. DOI: 10.48550/arXiv.2309.08793. arXiv: 2309.08793[cs.AI]. URL: <http://arxiv.org/abs/2309.08793> (visited on 07/21/2026).
- [110] Mark Rothermel et al. *VeriTaS: The First Dynamic Benchmark for Multimodal Automated Fact-Checking*. Apr. 30, 2026. DOI: 10.48550/arXiv.2601.08611. arXiv: 2601.08611[cs.IR]. URL: <http://arxiv.org/abs/2601.08611> (visited on 07/08/2026).
- [111] A. S. M. Sharifuzzaman Sagar et al. “Fact or Fake? Assessing the Role of Deepfake Detectors in Multimodal Misinformation Detection”. In: *IEEE Access* 14 (2026), pp. 86992–87011. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2026.3686437. URL: <https://ieeexplore.ieee.org/document/11493433> (visited on 08/13/2026).
- [112] Timo Schick et al. “Toolformer: language models can teach themselves to use tools”. In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS ’23. Red Hook, NY, USA: Curran Associates Inc., Dec. 10, 2023, pp. 68539–68551. (Visited on 08/27/2026).
- [113] Michael Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. “AVeriTeC: A Dataset for Real-world Claim Verification with Evidence from the Web”. In: *Advances in Neural Information Processing Systems* 36 (Dec. 15, 2023), pp. 65128–65167. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/cd86a30526cd1aff61d6f89f107634e4-Abstract-Datasets_and_Benchmarks.html (visited on 09/18/2025).
- [114] John Schulman et al. *Proximal Policy Optimization Algorithms*. Aug. 28, 2017. DOI: 10.48550/arXiv.1707.06347. arXiv: 1707.06347[cs.LG]. URL: <http://arxiv.org/abs/1707.06347> (visited on 07/30/2026).
- [115] Shaden Shaar et al. “That is a Known Lie: Detecting Previously Fact-Checked Claims”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, July 2020, pp. 3607–3618. DOI: 10.18653/v1/2020.acl-main.332. URL: <https://aclanthology.org/2020.acl-main.332/> (visited on 07/23/2026).
- [116] Zeinab Shahbazi and Sara Behnamian. “Using Large Language Models to Detect and Debunk Climate Change Misinformation”. In: *Big Data and Cognitive Computing* 10.1 (Jan. 2026), p. 34. ISSN: 2504-2289. DOI: 10.3390/bdcc10010034. URL: <https://www.mdpi.com/2504-2289/10/1/34> (visited on 06/30/2026).

-
- [117] Zhihong Shao et al. *DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models*. Apr. 27, 2024. DOI: 10.48550/arXiv.2402.03300. arXiv: 2402.03300[cs.CL]. URL: <http://arxiv.org/abs/2402.03300> (visited on 07/30/2026).
- [118] Ming Shen et al. *Optimizing LLM-Based Multi-Agent System with Textual Feedback: A Case Study on Software Development*. Aug. 6, 2025. DOI: 10.48550/arXiv.2505.16086. arXiv: 2505.16086[cs]. URL: <http://arxiv.org/abs/2505.16086> (visited on 10/07/2025).
- [119] Yongliang Shen et al. “HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face”. In: *Thirty-seventh Conference on Neural Information Processing Systems*. Nov. 2, 2023. URL: <https://openreview.net/forum?id=yHdTscY6Ci> (visited on 08/26/2026).
- [120] Noah Shinn et al. “Reflexion: language agents with verbal reinforcement learning”. In: *Advances in Neural Information Processing Systems*. Vol. 36. Curran Associates, Inc., 2023, pp. 8634–8652. DOI: 10.52202/075280-0377. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html (visited on 08/24/2026).
- [121] Charlie Victor Snell et al. “Scaling LLM Test-Time Compute Optimally Can be More Effective than Scaling Parameters for Reasoning”. In: *The Thirteenth International Conference on Learning Representations*. Oct. 4, 2024. URL: <https://openreview.net/forum?id=4FWAwZtd2n> (visited on 08/27/2026).
- [122] Elias Stengel-Eskin, Archiki Prasad, and Mohit Bansal. “REGAL: refactoring programs to discover generalizable abstractions”. In: *Proceedings of the 41st International Conference on Machine Learning*. Vol. 235. ICML’24. Vienna, Austria: JMLR.org, July 21, 2024, pp. 46605–46624. (Visited on 08/24/2026).
- [123] Yang Sui et al. “Stop Overthinking: A Survey on Efficient Reasoning for Large Language Models”. In: *Transactions on Machine Learning Research* (Apr. 5, 2025). ISSN: 2835-8856. URL: <https://openreview.net/forum?id=HvoG8SxggZ> (visited on 08/27/2026).
- [124] Zhen Sun et al. “Are We in the AI-Generated Text World Already? Quantifying and Monitoring AIGT on Social Media”. In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2025. Ed. by Wanxiang Che et al. Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 22975–23005. ISBN: 979-8-89176-251-0. DOI: 10.18653/v1/2025.acl-long.1120. URL: <https://aclanthology.org/2025.acl-long.1120/> (visited on 08/24/2026).
- [125] Dídac Surís, Sachit Menon, and Carl Vondrick. “ViperGPT: Visual Inference via Python Execution for Reasoning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 11888–11898. URL: https://openaccess.thecvf.com/content/ICCV2023/html/Suris_ViperGPT_Visual_Inference_via_Python_Execution_for_Reasoning_ICCV_2023_paper.html (visited on 08/24/2026).
- [126] Mirac Suzgun et al. “Dynamic Cheatsheet: Test-Time Learning with Adaptive Memory”. In: *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*. EACL 2026. Ed. by Vera Demberg, Kentaro Inui, and Lluís Marquez. Rabat, Morocco: Association for Computational Linguistics, Mar. 2026, pp. 7080–7106. ISBN: 979-8-89176-380-7. DOI: 10.18653/v1/2026.eacl-long.333. URL: <https://aclanthology.org/2026.eacl-long.333/> (visited on 07/21/2026).

-
- [127] Shreyas Kumar Tah et al. “HybridNet: Efficient Multimodal Fake News Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2026, pp. 11004–11012. URL: https://openaccess.thecvf.com/content/CVPR2026W/PPMisDet/html/Tah_HybridNet_Efficient_Multimodal_Fake_News_Detection_CVPRW_2026_paper.html (visited on 06/22/2026).
- [128] Gemma Team et al. *Gemma 4 Technical Report*. July 24, 2026. DOI: 10.48550/arXiv.2607.02770. arXiv: 2607.02770[cs.CL]. URL: <http://arxiv.org/abs/2607.02770> (visited on 08/14/2026).
- [129] Kimi Team et al. *Kimi k1.5: Scaling Reinforcement Learning with LLMs*. June 3, 2025. DOI: 10.48550/arXiv.2501.12599. arXiv: 2501.12599[cs.AI]. URL: <http://arxiv.org/abs/2501.12599> (visited on 07/03/2026).
- [130] James Thorne and Andreas Vlachos. “Automated Fact Checking: Task Formulations, Methods and Future Directions”. In: *Proceedings of the 27th International Conference on Computational Linguistics*. COLING 2018. Ed. by Emily M. Bender, Leon Derczynski, and Pierre Isabelle. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 3346–3359. URL: <https://aclanthology.org/C18-1283/> (visited on 06/28/2026).
- [131] James Thorne et al. “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, Volume 1 (Long Papers)*. NAACL-HLT 2018. Ed. by Marilyn Walker, Heng Ji, and Amanda Stent. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 809–819. DOI: 10.18653/v1/N18-1074. URL: <https://aclanthology.org/N18-1074/> (visited on 07/21/2026).
- [132] Jonathan Tonglet, Gabriel Thiem, and Iryna Gurevych. “COVE: COntext and VERacity prediction for out-of-context images”. In: *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. NAACL-HLT 2025. Ed. by Luis Chiruzzo, Alan Ritter, and Lu Wang. Albuquerque, New Mexico: Association for Computational Linguistics, Apr. 2025, pp. 2029–2049. ISBN: 979-8-89176-189-6. DOI: 10.18653/v1/2025.naacl-long.102. URL: <https://aclanthology.org/2025.naacl-long.102/> (visited on 08/24/2026).
- [133] Miles Turpin et al. “Language models don’t always say what they think: unfaithful explanations in chain-of-thought prompting”. In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS ’23. Red Hook, NY, USA: Curran Associates Inc., Dec. 10, 2023, pp. 74952–74965. (Visited on 07/06/2026).
- [134] Juraj Vladika, Phillip Schneider, and Florian Matthes. “HealthFC: Verifying Health Claims with Evidence-Based Medical Fact-Checking”. In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. LREC-COLING 2024. Ed. by Nicoletta Calzolari et al. Torino, Italia: ELRA and ICCL, May 2024, pp. 8095–8107. URL: <https://aclanthology.org/2024.lrec-main.709/> (visited on 07/21/2026).
- [135] Soroush Vosoughi, Deb Roy, and Sinan Aral. “The spread of true and false news online”. In: *Science* 359.6380 (Mar. 9, 2018), pp. 1146–1151. DOI: 10.1126/science.aap9559. URL: <https://www.science.org/doi/10.1126/science.aap9559> (visited on 08/25/2026).
- [136] Ivan Vykopal et al. *Generative Large Language Models in Automated Fact-Checking: A Survey*. arXiv.org. July 2, 2024. URL: <https://arxiv.org/abs/2407.02351v3> (visited on 07/01/2026).

-
- [137] Guanzhi Wang et al. “Voyager: An Open-Ended Embodied Agent with Large Language Models”. In: *Transactions on Machine Learning Research* (Nov. 29, 2023). ISSN: 2835-8856. URL: <https://openreview.net/forum?id=ehfRiF0R3a> (visited on 08/24/2026).
- [138] Lei Wang et al. “A survey on large language model based autonomous agents”. In: *Frontiers of Computer Science* 18.6 (Mar. 22, 2024), p. 186345. ISSN: 2095-2236. DOI: 10.1007/s11704-024-40231-1. URL: <https://doi.org/10.1007/s11704-024-40231-1> (visited on 07/08/2026).
- [139] Shengkang Wang et al. *MFC-Bench: Benchmarking Multimodal Fact-Checking with Large Vision-Language Models*. Feb. 17, 2025. DOI: 10.48550/arXiv.2406.11288. arXiv: 2406.11288[cs.CL]. URL: <http://arxiv.org/abs/2406.11288> (visited on 07/21/2026).
- [140] William Yang Wang. ““Liar, Liar Pants on Fire”: A New Benchmark Dataset for Fake News Detection”. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. ACL 2017. Ed. by Regina Barzilay and Min-Yen Kan. Vancouver, Canada: Association for Computational Linguistics, July 2017, pp. 422–426. DOI: 10.18653/v1/P17-2067. URL: <https://aclanthology.org/P17-2067/> (visited on 07/21/2026).
- [141] Xuezhi Wang et al. “Self-Consistency Improves Chain of Thought Reasoning in Language Models”. In: *The Eleventh International Conference on Learning Representations*. Sept. 29, 2022. URL: <https://openreview.net/forum?id=1PL1NIMMrw> (visited on 08/27/2026).
- [142] Yue Wang et al. *Thoughts Are All Over the Place: On the Underthinking of o1-Like LLMs*. Feb. 18, 2025. DOI: 10.48550/arXiv.2501.18585. arXiv: 2501.18585[cs.CL]. URL: <http://arxiv.org/abs/2501.18585> (visited on 07/06/2026).
- [143] Zihao Wang et al. “Describe, Explain, Plan and Select: Interactive Planning with LLMs Enables Open-World Multi-Task Agents”. In: *Advances in Neural Information Processing Systems*. Vol. 36. Curran Associates, Inc., 2023, pp. 34153–34189. DOI: 10.52202/075280-1480. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/6b8dfb8c0c12e6fafc6c256cb08a5ca7-Abstract-Conference.html (visited on 08/26/2026).
- [144] Zora Zhiruo Wang, Graham Neubig, and Daniel Fried. “TROVE: inducing verifiable and efficient toolboxes for solving programmatic tasks”. In: *Proceedings of the 41st International Conference on Machine Learning*. Vol. 235. ICML’24. Vienna, Austria: JMLR.org, July 21, 2024, pp. 51177–51191. (Visited on 08/24/2026).
- [145] Zora Zhiruo Wang et al. “Agent workflow memory”. In: *Proceedings of the 42nd International Conference on Machine Learning*. Vol. 267. ICML’25. Vancouver, Canada: JMLR.org, July 13, 2025, pp. 63897–63911. (Visited on 08/24/2026).
- [146] Guangyu Wei et al. *Towards Real-Time Fake News Detection under Evidence Scarcity*. Oct. 13, 2025. DOI: 10.48550/arXiv.2510.11277. arXiv: 2510.11277[cs.CL]. URL: <http://arxiv.org/abs/2510.11277> (visited on 06/22/2026).
- [147] Jason Wei et al. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”. In: *Advances in Neural Information Processing Systems*. Vol. 35. Curran Associates, Inc., 2022, pp. 24824–24837. DOI: 10.52202/068431-1800. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract.html (visited on 08/27/2026).
- [148] Junjie Wu et al. *E2LVLM: Evidence-Enhanced Large Vision-Language Model for Multimodal Out-of-Context Misinformation Detection*. Feb. 12, 2025. DOI: 10.48550/arXiv.2502.10455. arXiv: 2502.10455[cs.LG]. URL: <http://arxiv.org/abs/2502.10455> (visited on 06/30/2026).

-
- [149] Shirley Wu et al. “AvaTaR: Optimizing LLM Agents for Tool Usage via Contrastive Reasoning”. In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. Nov. 6, 2024. URL: <https://openreview.net/forum?id=N4quRxEl9p> (visited on 08/26/2026).
- [150] Tongtong Wu et al. *Continual Learning for Large Language Models: A Survey*. Feb. 7, 2024. DOI: 10.48550/arXiv.2402.01364. arXiv: 2402.01364[cs.CL]. URL: <http://arxiv.org/abs/2402.01364> (visited on 07/17/2026).
- [151] Yin Wu et al. *EXCLAIM: An Explainable Cross-Modal Agentic System for Misinformation Detection with Hierarchical Retrieval*. Mar. 1, 2025. DOI: 10.48550/arXiv.2504.06269. arXiv: 2504.06269[cs.IR]. URL: <http://arxiv.org/abs/2504.06269> (visited on 06/30/2026).
- [152] Yuzhuo Xiao et al. *XFacta: Contemporary, Real-World Dataset and Evaluation for Multimodal Misinformation Detection with Multimodal LLMs*. Aug. 4, 2025. DOI: 10.48550/arXiv.2508.09999. arXiv: 2508.09999[cs]. URL: <http://arxiv.org/abs/2508.09999> (visited on 09/23/2025).
- [153] Miao Xiong et al. “Can LLMs Express Their Uncertainty? An Empirical Evaluation of Confidence Elicitation in LLMs”. In: *The Twelfth International Conference on Learning Representations*. Oct. 13, 2023. URL: <https://openreview.net/forum?id=gjeQKFxFpZ> (visited on 08/24/2026).
- [154] Qian Xiong et al. “Butterfly Effects in Toolchains: A Comprehensive Analysis of Failed Parameter Filling in LLM Tool-Agent Systems”. In: *Findings of the Association for Computational Linguistics: EMNLP 2025*. Findings 2025. Ed. by Christos Christodoulopoulos et al. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 16712–16729. ISBN: 979-8-89176-335-7. DOI: 10.18653/v1/2025.findings-emnlp.907. URL: <https://aclanthology.org/2025.findings-emnlp.907/> (visited on 08/26/2026).
- [155] Danni Xu et al. *RW-Post: Auditable Evidence-Grounded Multimodal Fact-Checking in the Wild*. May 12, 2026. DOI: 10.48550/arXiv.2512.22933. arXiv: 2512.22933[cs.AI]. URL: <http://arxiv.org/abs/2512.22933> (visited on 06/22/2026).
- [156] Wenyan Xu et al. *MMM-Fact: A Multimodal, Multi-Domain Fact-Checking Dataset with Multi-Level Retrieval Difficulty*. Oct. 29, 2025. DOI: 10.48550/arXiv.2510.25120. arXiv: 2510.25120[cs.SI]. URL: <http://arxiv.org/abs/2510.25120> (visited on 07/21/2026).
- [157] Keyang Xuan et al. *LEMMA: Towards LVLM-Enhanced Multimodal Misinformation Detection with External Knowledge Augmentation*. June 20, 2024. DOI: 10.48550/arXiv.2402.11943. arXiv: 2402.11943[cs]. URL: <http://arxiv.org/abs/2402.11943> (visited on 12/23/2025).
- [158] Bingjian Yang et al. “A New Dataset and Benchmark for Grounding Multimodal Misinformation”. In: *Proceedings of the 33rd ACM International Conference on Multimedia*. Oct. 27, 2025, pp. 12571–12577. DOI: 10.1145/3746027.3758191. arXiv: 2509.08008[cs.SI]. URL: <http://arxiv.org/abs/2509.08008> (visited on 07/21/2026).
- [159] Chengrun Yang et al. “Large Language Models as Optimizers”. In: *International Conference on Learning Representations 2024* (May 31, 2024), pp. 12028–12068. URL: https://proceedings.iclr.cc/paper_files/paper/2024/hash/3339f19c5fcee3ad74502947a32be9e6-Abstract-Conference.html (visited on 08/24/2026).
- [160] Shuo Yang et al. “RAMA: Retrieval-Augmented Multi-Agent Framework for Misinformation Detection in Multimodal Fact-Checking”. In: *Companion Proceedings of the ACM Web Conference 2026*. WWW Companion ’26. New York, NY, USA: Association for Computing Machinery, May 28, 2026, pp. 416–424. ISBN: 979-8-4007-2308-7. DOI: 10.1145/3774905.3796483. URL: <https://dl.acm.org/doi/10.1145/3774905.3796483> (visited on 08/24/2026).

-
- [161] Barry M. Yao et al. *End-to-End Multimodal Fact-Checking and Explanation Generation: A Challenging Dataset and Models (MOCHEG)*. July 18, 2023. URL: <https://dl.acm.org/doi/abs/10.1145/3539618.3591879> (visited on 09/18/2025).
- [162] Shunyu Yao et al. “ReAct: Synergizing Reasoning and Acting in Language Models”. In: *The Eleventh International Conference on Learning Representations*. Sept. 29, 2022. URL: https://openreview.net/forum?id=WE_vluYUL-X (visited on 08/27/2026).
- [163] Shunyu Yao et al. “Tree of Thoughts: Deliberate Problem Solving with Large Language Models”. In: *Advances in Neural Information Processing Systems*. Vol. 36. Curran Associates, Inc., 2023, pp. 11809–11822. DOI: 10.52202/075280-0517. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract.html (visited on 08/27/2026).
- [164] Andrii Yermakov et al. “Deepfake Detection that Generalizes Across Benchmarks”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2026, pp. 773–783. URL: https://openaccess.thecvf.com/content/WACV2026/html/Yermakov_Deepfake_Detection_that_Generalizes_Across_Benchmarks_WACV_2026_paper.html (visited on 07/29/2026).
- [165] Gal Yona, Roei Aharoni, and Mor Geva. “Can Large Language Models Faithfully Express Their Intrinsic Uncertainty in Words?”. In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2024. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 7752–7764. DOI: 10.18653/v1/2024.emnlp-main.443. URL: <https://aclanthology.org/2024.emnlp-main.443/> (visited on 08/13/2026).
- [166] Fei Yu et al. “Natural Language Reasoning, A Survey”. In: *ACM Computing Surveys* 56.12 (Oct. 3, 2024), 304:1–304:39. ISSN: 0360-0300. DOI: 10.1145/3664194. URL: <https://dl.acm.org/doi/10.1145/3664194> (visited on 08/24/2026).
- [167] Mert Yuksekgonul et al. “Optimizing generative AI by backpropagating language model feedback”. In: *Nature* 639.8055 (Mar. 2025), pp. 609–616. ISSN: 1476-4687. DOI: 10.1038/s41586-025-08661-4. URL: <https://www.nature.com/articles/s41586-025-08661-4> (visited on 10/09/2025).
- [168] Qiusi Zhan et al. “InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents”. In: *Findings of the Association for Computational Linguistics: ACL 2024*. Findings 2024. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 10471–10506. DOI: 10.18653/v1/2024.findings-acl.624. URL: <https://aclanthology.org/2024.findings-acl.624/> (visited on 08/26/2026).
- [169] Fanrui Zhang et al. *Fact-R1: Towards Explainable Video Misinformation Detection with Deep Reasoning*. Oct. 15, 2025. DOI: 10.48550/arXiv.2505.16836. arXiv: 2505.16836 [cs.CV]. URL: <http://arxiv.org/abs/2505.16836> (visited on 06/22/2026).
- [170] Fanrui Zhang et al. “Navigating Truth in Multimodal Fact-checking via Retrieval- and Reasoning-Enhanced Large Language Models”. In: *Proceedings of the ACM Web Conference 2026*. ACM Conferences. Apr. 13, 2026, pp. 1841–1851. ISBN: 979-8-4007-2307-0. DOI: 10.1145/3774904.3792706. URL: <https://dl.acm.org/doi/abs/10.1145/3774904.3792706> (visited on 06/22/2026).
- [171] Jiayi Zhang et al. “AFlow: Automating Agentic Workflow Generation”. In: *The Thirteenth International Conference on Learning Representations*. Oct. 4, 2024. URL: <https://openreview.net/forum?id=z5uVAKwmjf> (visited on 08/24/2026).
- [172] Qizheng Zhang et al. “Agentic Context Engineering: Evolving Contexts for Self-Improving Language Models”. In: *The Fourteenth International Conference on Learning Representations*. Oct. 8, 2025. URL: <https://openreview.net/forum?id=eC4ygDs02R> (visited on 08/26/2026).

-
- [173] Wenqi Zhang et al. “Agent-Pro: Learning to Evolve via Policy-Level Reflection and Optimization”. In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2024. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 5348–5375. DOI: 10.18653/v1/2024.acl-long.292. URL: <https://aclanthology.org/2024.acl-long.292/> (visited on 08/24/2026).
- [174] Andrew Zhao et al. “ExpEL: LLM agents are experiential learners”. In: *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*. Vol. 38. AAAI’24/IAAI’24/EAAI’24. AAAI Press, Feb. 20, 2024, pp. 19632–19642. ISBN: 978-1-57735-887-9. DOI: 10.1609/aaai.v38i17.29936. URL: <https://doi.org/10.1609/aaai.v38i17.29936> (visited on 08/24/2026).
- [175] Yingming Zheng et al. “CRAVE: A Conflicting Reasoning Approach for Explainable Claim Verification Using LLMs”. In: *2025 International Joint Conference on Neural Networks (IJCNN)*. 2025 International Joint Conference on Neural Networks (IJCNN). ISSN: 2161-4407. June 2025, pp. 1–8. DOI: 10.1109/IJCNN64981.2025.11227412. URL: <https://ieeexplore.ieee.org/document/11227412> (visited on 08/24/2026).
- [176] Andy Zhou et al. “Language Agent Tree Search Unifies Reasoning Acting and Planning in Language Models”. In: *Proceedings of the 41st International Conference on Machine Learning*. Vol. 235. ICML’24. Vienna, Austria: JMLR.org, July 21, 2024, pp. 62138–62160. (Visited on 08/26/2026).
- [177] Jianpeng Zhou et al. “Adaptive-solver framework for dynamic strategy selection in large language model reasoning”. In: *Information Processing & Management* 62.3 (May 1, 2025), p. 104052. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2024.104052. URL: <https://www.sciencedirect.com/science/article/pii/S0306457324004114> (visited on 12/17/2025).
- [178] Yongchao Zhou et al. “Large Language Models are Human-Level Prompt Engineers”. In: *The Eleventh International Conference on Learning Representations*. Sept. 29, 2022. URL: <https://openreview.net/forum?id=92gvk82DE-> (visited on 08/24/2026).
- [179] Ziyu Zhuang. *PASS-FC: Progressive and Adaptive Search Scheme for Fact Checking of Comprehensive Claims*. May 26, 2025. DOI: 10.48550/arXiv.2504.09866. arXiv: 2504.09866[cs]. URL: <http://arxiv.org/abs/2504.09866> (visited on 12/10/2025).
- [180] Wei Zou et al. “PoisonedRAG: knowledge corruption attacks to retrieval-augmented generation of large language models”. In: *Proceedings of the 34th USENIX Conference on Security Symposium*. SEC’25. USA: USENIX Association, Aug. 13, 2025, pp. 3827–3844. ISBN: 978-1-939133-52-6. (Visited on 08/27/2026).

Use of AI Tools

Generative AI tools, primarily Claude by Anthropic, were used to support the work in this thesis. Their use fell into two areas. In implementation, these tools assisted with writing, debugging, and refactoring code for the fact-checking pipeline and the analysis scripts. In writing, they assisted with language editing, rephrasing, and improving the structure and clarity of the text.

All AI-assisted output was reviewed, tested, and where necessary corrected by the author. The research questions, experimental design, analysis, and conclusions are the author's own, and the author takes full responsibility for the content of this thesis, including any parts drafted or edited with AI support.

A. Appendix

A.1. Configuration Studies

This appendix records the studies that fixed the configuration of Table 5.3. None is a contribution, and each is reported here so that Chapter 5 can state a decision without arguing for it. The system has more configurable parts than the main comparison can vary. Each of the studies in this section settles one of them, and together they define the configuration every later experiment holds fixed. Several are reported at length elsewhere, and this section states the decision each one produced and points to the full treatment.

Two conventions apply throughout. Every comparison is paired on claims, since the benchmark’s difficulty varies enough between subsamples to swamp the effects under test. Effects are reported as relative movement within a comparison rather than as absolute error, because the pipeline was under active development while these studies ran. A change measured against one build is not comparable to a change measured against another, and the final absolute numbers appear only in section 5.2.

A.1.1. Judge Configuration

The judge maps an evidence set onto the benchmark’s seven-point integrity scale. Three settings were fixed experimentally. Resampling the judge on an identical prompt moves squared error by about as much as most of the prompt edits we considered, and swaps a fifth of the direction flips from one sample to the next. Judge-side comparisons below that magnitude are therefore not interpretable at all from a single run. Averaging three independent samples numerically removes roughly a tenth of the error and about a third of the direction flips relative to a single sample, at a small fraction of the per-claim budget, since the judge call is far cheaper than the investigation that precedes it. We use three samples with mean-numeric aggregation. We also tested decomposing the verdict into two calls, one for direction and one for certainty. The decomposition drifts certainty toward the middle of the scale and loses accuracy on the seven-point label without improving squared error, so the judge emits the seven-way label directly. The judge is systematically overconfident relative to its own direction accuracy, which makes certainty an obvious target. We do not calibrate. A global shrinkage of confident predictions does improve squared error, but it does so by exploiting the benchmark’s fixed label-to-value mapping rather than by knowing anything more about the claim, and it leaves every verdict unchanged. A learned per-claim calibrator was built and abandoned. Appendix A.4 reports it in full, including the imbalance artifact that made an early version look successful.

A.1.2. Evidence Budget and Presentation

Each delegated search task runs at most two rounds, issues at most three queries per round, and retrieves at most three sources per query. Raising the budget is not the binding constraint. Most of what additional retrieval returns is on topic without bearing on the verdict, and the deficit lies in which sources are surfaced and selected rather than in how many are read. Section 5.3.3 quantifies that split. How evidence reaches the judge mattered more than how much of it there is. Two presentation decisions follow from the error analysis. Sources that could not be fetched are labeled as such instead of being passed off as short summaries, so the judge can tell a thin reading from a complete one. Reverse image search results are promoted to individual evidence items only when the search reports a genuine match, because an unfiltered fan-out lets a single tool call present one weak lookalike set as many independent corroborations. The second fix removes a specific and recurring failure in which the absence of established provenance was synthesized into a positive finding of miscontextualization.

A.1.3. Blueprints

The generation pipeline of Section 4.4 reached its final form through a sequence of failures, each of which added one guard. We record them here because the guards are configuration, and because the first fully automatic pool was worse than the hand-curated one it was meant to replace.

Concentration and Consolidation. One broadly worded blueprint absorbed close to half of all traffic as a catch-all, and it accounted for more than the entire deficit. The hand-curated pool had avoided this only by accident, since its independently written descriptions happened to spread the selector's choices across several deeper blueprints. Nothing in the pipeline had enforced it. Oversized clusters are now split recursively and their shards repacked under a size cap, representatives are drawn to expose heterogeneity rather than to summarize a centroid, and a split recommendation is acted on rather than recorded. The merge stage then re-created catch-alls by another route, so merges whose combined training share exceeds the cap are vetoed. A merged blueprint inherits the larger of its parents' iteration budgets, and any blueprint carrying a substantial share of traffic receives a floor on its budget. The size cap was removed once its premise no longer held. Diagnosis of the dominant blueprint showed that its checklist was effectively unsatisfiable, and that unresolved checks were being read as evidence of guilt, with error rising steeply in the number of checks left open. That is a property of over-specified blueprints rather than of large ones, and the routing controls had by then shown that the distribution of traffic across a pool is not what drives the error. Removing the cap yields the compact pool of four specialized blueprints plus a generic fallback used in every experiment that follows. Larger blueprint pools consistently led to worse performance in otherwise identical settings.

Verification stance. A blueprint must not presuppose the pathology it is named for. The worst performer in the first pool described its own target as authentic media shared in false context and instructed the graph to expose the mismatch, so any lookalike match was read as the hunted mismatch and authentic media was flipped to compromised. Rewording it neutrally, with symmetric outcomes and a required check that a match refers to this media, cut its error by about a fifth and removed most of those flips. The stance is now enforced at authoring time for every blueprint, by rules in the synthesis prompt and by a lint over instruction-bearing fields that triggers one automatic rewording pass. A pool-wide contrast pass follows, since the selector reads descriptions and mutually exclusive descriptions are what make its job well posed.

Planner	Method	Calls	In tok	Out tok	Wall clock	Cost
GEMINI 3 FLASH	Blueprints	32,208	303.2M	7.98M	9.3 h	\$175
	Single	34,939	250.0M	8.45M	5.2 h	\$150
	Empty	32,388	214.0M	7.73M	4.7 h	\$130
GEMMA 4 31B	Blueprints	34,024	150.2M	5.79M	7.3 h	self-hosted
	Single	18,880	73.2M	2.99M	5.8 h	self-hosted
	Empty	18,138	57.1M	2.74M	5.5 h	self-hosted

Table A.1.: Measured cost and runtime of the six reported runs. Token counts and call counts come from the run traces. Wall clock is the union of the per-claim execution intervals, which excludes the gaps introduced by resuming an interrupted run. Dollar figures cover model calls alone. Claim counts fall short of 1000 where a claim failed to score.

A.2. Cost and Runtime

Table A.1 records what one execution over the 1000 evaluation claims consumes, measured from the traces of the six runs reported in Chapter 5. A run issues between 18,000 and 35,000 model calls and processes 57M to 303M input tokens, since every retrieved page is summarized in a call of its own. Claims execute in parallel, which is why a run completes in 4.7 to 9.3 hours while a single claim takes 1.4 to 4.4 minutes end to end. In-pipeline judging is a small part of the total, at exactly three calls per claim and 9 to 19% of a run’s input tokens. Rescoring a finished run with the remaining four judges costs roughly as much again as producing it, which is the price of the panel Section 5.1.2 argues for.

A.3. Tool Analyses

A.3.1. Geo-Location Tool Analysis

This appendix documents the comparison that motivated the choice of a general-purpose LLM over a specialized geo-location model in Section 4.3.4.

Datasets. Two evaluation sets are used, differing in provenance for a deliberate reason. The first consists of 238 personally collected photographs that have never been published online, each labeled with the country in which it was taken. Because these images cannot occur in any model’s training data, they isolate genuine visual geolocation ability from recall of a previously seen image. The second is derived from the VERITAS dataset and consists of 95 images taken from fact-checked claims. Their labels are extracted from the debunking articles by an LLM under a strict-extraction protocol: a country is recorded only when the article states it explicitly, and every label must be accompanied by a verbatim quote from the article that names the location, so that each label remains auditable against its source. Images are retained only if the underlying photograph is genuine and unaltered, and only if the scene is geolocatable from visual content alone; broadcast stills, studio interviews, screenshots, and synthetic images are excluded, since the country of an event is frequently known while the image itself carries no location signal. This set is closer to the distribution the tool actually encounters in the pipeline, at the cost of consisting entirely of images that circulated publicly.

Method	Self-collected ($n = 233$)		VERITAS ($n = 95$)	
	Top-1	Top-3	Top-1	Top-3
STREETCLIP [49]	53.7%	73.0%	46.3%	65.3%
GEOCLIP [20]	58.8%	83.3%	56.8%	67.4%
CLAUDE OPUS 4.8	83.7%	94.0%	85.3%	91.6%
GEMINI 3 FLASH	85.4%	95.3%	85.3%	91.6%

Table A.2.: Country-level geolocation accuracy in percent.

Methods and metrics. Four methods are compared. STREETCLIP [49] is applied zero-shot: the image and a fixed list of 91 candidate country names are embedded jointly and the three highest-scoring countries are taken. GEOCLIP [20] predicts GPS coordinates rather than countries; its top 100 coordinate predictions are therefore reverse-geocoded offline and their probability mass is summed per country, which is what makes a top-3 country list meaningful, as the raw top-3 coordinates are typically three points in the same country. GEMINI 3 FLASH and CLAUDE OPUS 4.8 are given the image and asked for their three most likely countries in a fixed output format, without explanation. Both top-1 and top-3 country accuracy are reported. Predictions and labels are canonicalized to ISO 3166-1 alpha-2 codes before comparison. Accuracies are computed over the images for which every method produced a prediction (233 of 238 and 95 of 95 respectively).

Results. Both LLMs outperform both specialized models by a wide margin on both datasets, and the ordering is stable across them. On the self-collected set, GEMINI 3 FLASH is correct on 78 images on which STREETCLIP fails while failing on only 4 on which STREETCLIP succeeds. Against GEOCLIP the split is 68 to 6. The gap widens on the VERITAS-derived set, where it reaches 39 percentage points against STREETCLIP (39 to 2). The difference between the two LLMs, in contrast, is not significant on either set, so the choice between them rests on cost rather than on accuracy.

Contamination Because the VERITAS-derived images all circulated publicly and were discussed in fact-checking articles, an LLM might name their location from having encountered them during training rather than from the visual content. The self-collected set controls for this directly, as those photographs have never been published. GEMINI 3 FLASH reaches 85.41% top-1 accuracy on the private images against 85.26% on the public ones, which is effectively identical performance. The advantage over the specialized models is therefore not attributable to memorization of the evaluation images.

A.3.2. Manipulation Tool Analysis

This appendix evaluates the manipulation and AI-generation detectors on the images of the VERITAS 2026 Q1 split, that is, on the same distribution the pipeline encounters rather than on a forensic benchmark.

Ground truth

The dataset ships an `authenticity.score` per media file, but that field cannot serve as ground truth for a forensic detector. It blends direction with certainty, and it scores how *misleading* the file is rather than whether

the file was altered: a staged but pristine recording sits at -0.42 , and a fact-check that never examined provenance at all lands near 0, indistinguishable by score from a genuine borderline case.

We therefore derive a file-level label from the fact-checkers’ free-text authenticity justifications instead, using an LLM extraction pass over the justification prose:

manipulated the file itself is synthetic, or was altered after capture;

authentic the file is an unaltered capture, even if staged or miscaptioned;

unknown the fact-check did not assess provenance at all.

Of 493 images, 221 (44.8%) are manipulated, 152 (30.8%) authentic, and 120 (24.3%) unknown. The unknown rows are unlabelled data, not negatives, and are excluded from every number below. Notably, 82 of the 152 authentic files were nonetheless misleading, e.g. staged scenes or false captions. They count as authentic here, because staging is not manipulation, and a detector that fires on them is wrong.

Each manipulated file also carries a manipulation *type*, and each type a flag for whether a pixel-forensics detector could catch it even in principle. A cleanly rendered fake screenshot, for instance, contains no splice to find. This lets us report two views and separate “the detector is weak” from “the corpus is outside its design envelope”, which a single pooled number cannot do.

Detectors under test

C2PA provenance manifests cryptographically bound to the file at capture or generation time. Reported as coverage, never as AUC: a missing manifest is not a negative prediction, so ranking files by manifest absence is meaningless.

TruFor [47] a learned forgery localiser producing a per-image manipulation score from noise-residual and RGB features.

Sightengine a commercial API; we use its `genai` and `deepfake` heads, and their maximum as a combined score.

GenD [164] a face-centric deepfake detector, which scores only images containing a sufficiently large face.

Pixel detectors

Table A.3 gives the headline result. **TruFor is non-discriminative on this corpus** (AUC 0.545, and 0.530 even when restricted to manipulations it was designed to find). Its apparently high TPR of 0.93 is an artefact of a threshold that also fires on 79% of authentic files: it calls almost everything manipulated. Restricting to detectable cases does not rescue it, which locates the failure in the detector rather than in the corpus. Our reading is that TruFor’s noise-residual features do not survive the recompression, rescaling, and screenshotting that every file in a redistributed-media corpus has passed through.

Sightengine’s AI-generation head is the only usable signal. It reaches AUC 0.742 on detectable manipulations at a tolerable 9.2% false-positive rate, and it is the only detector with meaningful recall at high precision as it recovers half of detectable manipulations at 90% precision, where every other detector recovers essentially none. The `deepfake` head is much weaker in isolation (0.616), and the combined maximum (0.791) is driven mostly by the AI-generation component.

Detector	All manipulated (221 vs. 152)				Detectable only (184 vs. 152)			
	AUC	TPR	FPR	R@P=0.9	AUC	TPR	FPR	R@P=0.9
TruFOR	0.545	0.932	0.789	0.005	0.530	0.918	0.789	0.005
Sightengine ai_gen	0.712	0.493	0.092	0.448	0.742	0.549	0.092	0.500
Sightengine deepfake	0.584	0.593	0.447	0.009	0.616	0.647	0.447	0.011
Sightengine max	0.740	0.484	0.092	0.416	0.791	0.603	0.145	0.462

Table A.3.: Detector performance against the file-level integrity labels. The two column groups share the same 152 authentic files and differ only in the positive class: *all manipulated* pools every manipulated file, while *detectable only* keeps the 184 whose manipulation type a pixel detector could catch in principle, dropping the 37 cases that leave no pixel trace. The gap between the groups therefore separates a weak detector from a corpus outside its design envelope. Thresholds are per-detector Youden optima, and R@P=0.9 is recall at 90% precision. Since the authentic set and the threshold are both unchanged, FPR is identical across the groups for every detector except Sightengine max, whose optimal threshold shifts from 0.050 to 0.020 with the positive class.

Type	Detectable	n	TruFOR	ai_gen	deepfake
ai_generated	yes	115	0.913	0.687	0.548
graphic_edit	yes	39	0.897	0.410	0.744
fabricated_screenshot	partial	35	1.000	0.229	0.343
splice_composite	yes	15	1.000	0.200	0.867
deepfake	yes	15	0.933	0.200	0.933
other_manipulation	partial	2	1.000	0.000	0.000
<i>false-positive rate</i>		152	0.789	0.092	0.447

Table A.4.: Recall per manipulation type at each detector’s Youden threshold. The final row is the false-positive rate on the 152 authentic files, which is what these recalls cost.

Recall by manipulation type Table A.4 shows the two heads are complementary in the way their design predicts: ai_gen is strongest on wholesale synthesis (0.687), deepfake on face swaps (0.933) and splices (0.867). It also shows why TruFOR’s per-type recalls (1.000 on three categories) must not be read as strength because they come with a 0.789 false-positive rate, i.e. the same threshold fires on four out of five authentic images.

The hardest category is fabricated_screenshot ($n = 35$), where the best head recovers 0.229. This is expected rather than disappointing: a screenshot of a post that never existed is a faithful capture of a real rendering, so there is no pixel-level trace to find. Detecting it requires establishing that the underlying post does not exist, which is a retrieval problem, not a forensic one.

C2PA No image in the corpus carries a C2PA manifest. Every image here is a redistributed copy, i.e. screenhotted, re-encoded, or scraped from a platform, and all of those paths strip the manifest. This is worth stating precisely, because it is a property of the distribution channel rather than of C2PA. The standard works as designed at the point of capture; it simply carries no information by the time media reaches a fact-checker. The absence of a manifest is *not* evidence of manipulation: on this corpus it is uninformative about every file, authentic and manipulated alike. A pipeline that treated missing provenance as a negative signal would be inferring from a constant.

GenD GEND scores only images containing a sufficiently large face, and declines the rest. Of 493 images it scored 244 (65.4%), returning `no_face` on 108 (29.0%) and `faces_too_small` on 19 (5.1%). The coverage is not class-neutral with 72% on manipulated files versus 56% on authentic ones, since manipulated images contain faces more often. For this reason we compare all detectors on GEND’s own scored subset (159 manipulated vs. 85 authentic). On that paired subset GEND reaches **AUC 0.398, below chance**, including the categories it targets (0.461 on deepfakes, 0.402 on AI-generated). Its median score is *higher* for authentic images (0.524) than for any manipulation type. On the identical subset, Sightengine’s `ai_gen` head scores 0.721. Image resolution has been ruled out as that confound (Spearman $\rho = -0.07$). We conclude GEND does not transfer to this corpus, and it is implemented but not enabled in the pipeline.

A.4. Calibrating the Judge

LLMs’ verbalized confidence is not necessarily well calibrated with respect to the correctness of their predictions [69, 153, 165]. Our setting asks something different from the judge, namely how strongly the retrieved evidence supports a claim, independent of the model’s own subjective confidence. In experiments the judge usually got the direction right while confusing *certain* and *rather certain*, at a rate of 51% within that band. Prompt engineering did not remove the confusion. We therefore decouple the two quantities and let the LLM determine the direction of the evidence (🟢 **Intact** / 🟡 **Compromised**) while a separate regressor estimates its strength. For 🟡 **Unknown**, no additional certainty is predicted.

Setup. Ground truth is a continuous integrity score, and the 7 labels are a quantization of it. The task is thus ordinal regression on a latent score with the sign already solved. We keep the LLM judge unchanged for the direction and add a standalone regressor that predicts the magnitude $|\text{integrity.score}| \in [0, 1]$ from execution-trace features. The final prediction is the combined score

$$\text{verdict} = \text{sign}(\text{judge_direction}) \cdot \text{regressor_magnitude}.$$

We train a Histogram-based Gradient Boosting Regression Tree on 500 claims sampled from VERITAS Q2–Q4 2025, of which 493 yield a usable trace, and report 5-fold out-of-fold predictions. A run over 196 claims from Q1 2026 serves as a held-out test set.

Regressor Input. The input is composed of structured and embedding-derived features. Structured features include the direction predicted by the judge, evidence count, usefulness ratio of evidence, number of distinct domains, number of iterations, evidence growth, runtime, justification length, judge output tokens, number of media items in the claim, claim language, claim length and the gap between claim date and evidence dates. Embeddings are computed for evidence items using OpenAI’s `text-embedding-3-large`¹. The derived inputs then include evidence agreement as mean, max and variance of the pairwise cosine distances, on-topicality as the cosine distance between claim and evidence (mean and max), and justification semantics as the embedding norm plus the top 32 PCA components.

Results. Table A.5 shows the results against a baseline that uses pure LLM judging. Every metric moves in the intended direction, and the targeted confusion drops by more than half. A closer look at the predictions

¹<https://developers.openai.com/api/docs/models/text-embedding-3-large>

	MSE ($\downarrow$)	MAE ($\downarrow$)	ECE ($\downarrow$)	Acc. 7 ($\uparrow$)	Acc. 3 ($\uparrow$)	<i>certain</i> $\leftrightarrow$ <i>rather certain</i> confusion ($\downarrow$)
Judge (baseline)	0.254	0.292	0.178	44.0%	89.9%	51.1%
Always <i>certain</i>	0.275	0.231	0.096	70.0%	89.9%	22.8%
Regressor	0.244	0.204	0.007	69.6%	89.9%	23.0%

Table A.5.: Certainty calibration on 493 VERITAS 2025 claims, 5-fold out-of-fold. The trivial predictor that always answers *certain* matches the learned regressor on both headline metrics, which exposes the apparent gain over the judge as an artifact of class imbalance.

	MSE ($\downarrow$)	MAE ($\downarrow$)	Acc. 7 ($\uparrow$)	Acc. 3 ($\uparrow$)
Judge (baseline)	0.337	0.341	41.3%	85.7%
Always <i>certain</i>	0.429	0.306	63.8%	85.7%
Regressor	0.386	0.282	62.8%	85.7%

Table A.6.: Held-out generalization on 196 VERITAS Q1 2026 claims, using the model fit on the 2025 training set. The regressor loses to the constant predictor on accuracy and to the judge on MSE.

dissolves that win. The regressor’s output has a standard deviation of 0.027 where the truth has 0.124, and its mean prediction for truly *certain* claims (0.913) is indistinguishable from the one for *rather certain* claims (0.904). The model has learned a constant close to the base rate. A trivial predictor that always answers *certain* therefore matches it on 7-class accuracy (70.0% against 69.6%) and on the targeted confusion (22.8% against 23.0%). What remains of the MSE and ECE advantage comes from shrinking that constant from 1.0 to 0.91, which is regression toward the mean rather than per-claim discrimination. Macro-F1 makes the same point, moving from 0.235 to 0.234 while accuracy gains 25 points.

Held-out data confirms the collapse. Applied to 196 claims from Q1 2026, the regressor reaches 62.8% 7-class accuracy against 63.8% for the always-*certain* predictor, and its MSE of 0.386 is worse than the 0.337 of the plain judge (Table A.6). The behavior is model-agnostic. On the binary *certain* versus *rather certain* decision, logistic regression, gradient boosting and a multi-layer perceptron all reach cross-validated AUCs between 0.57 and 0.59 while fitting the training set perfectly, and the learning curve is flat beyond roughly 220 training claims. The ceiling is the information carried by the execution trace, not model capacity or sample size.

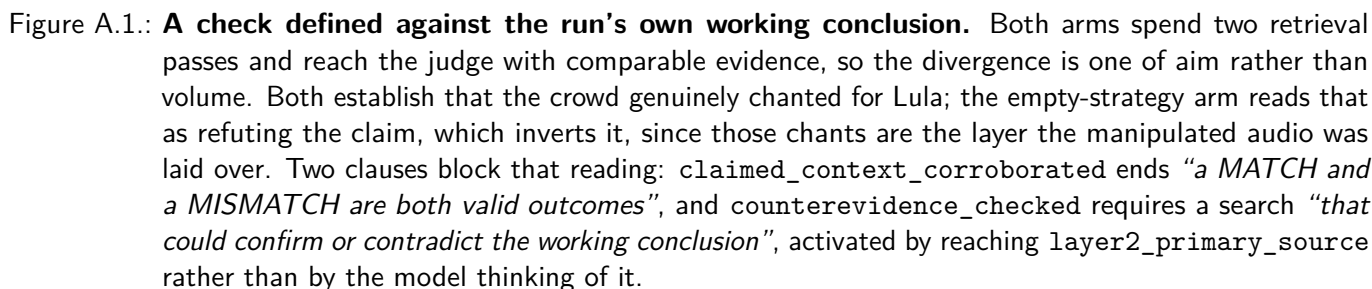
Class imbalance is what made the first reading look like a success. With 76% of the claims labeled *certain*, accuracy rewards the majority class, and a constant at the base rate is calibrated in aggregate, so ECE falls without any discrimination being learned. We consequently report the always-*certain* baseline next to every calibration number, and treat rank correlation rather than ECE as the decisive diagnostic. If the approach were revisited, the lever would be a different signal, such as the judge’s token-level logprobs or an explicit numeric confidence, since a different model class does not help.

A.5. Qualitative Sample

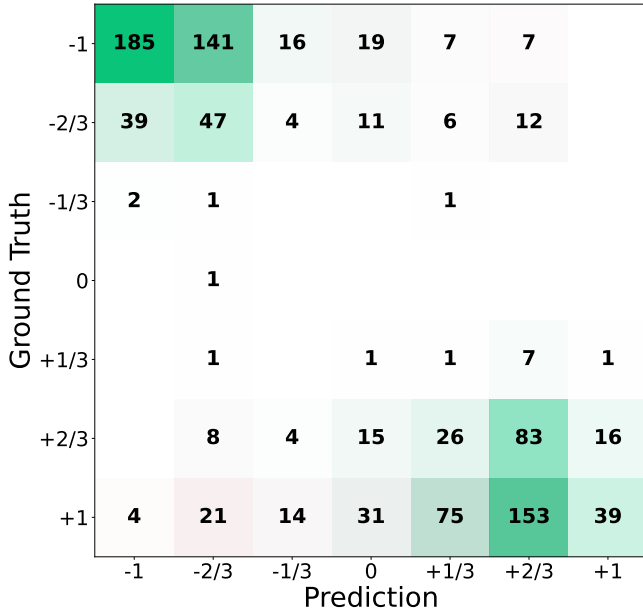
Figure A.1 is the second of the two presented cases where the blueprint’s check ledger changes the verdict, discussed in Section 5.2.2. It replicates the mechanism of Figure 5.5 on a different claim type, and is also the case where the delegated media task asks for an analysis the runtime cannot perform.

EN *The video shows a demonstration in Madrid in solidarity with Iran and against the attacks by Israel and the United States.*

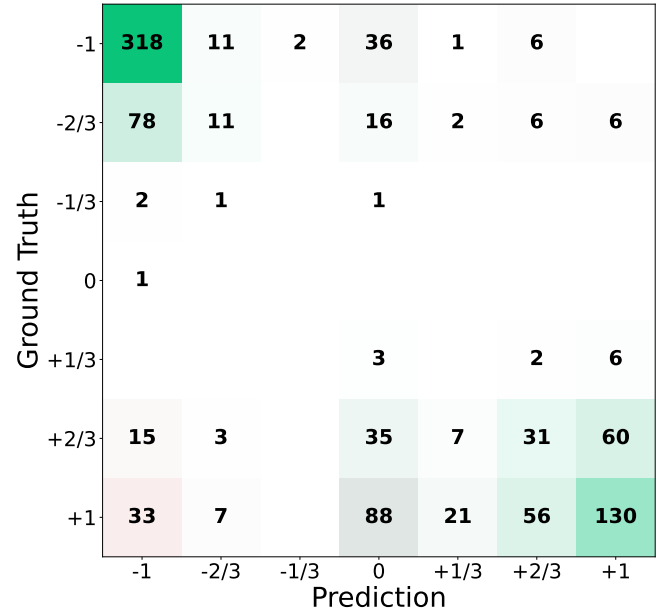
A large crowd of people is gathered on a hillside, looking towards the stadium. The crowd is dense and diverse in age and appearance. The hillside is covered in grass and some trees. The stadium lights are visible in the background, illuminating the scene.



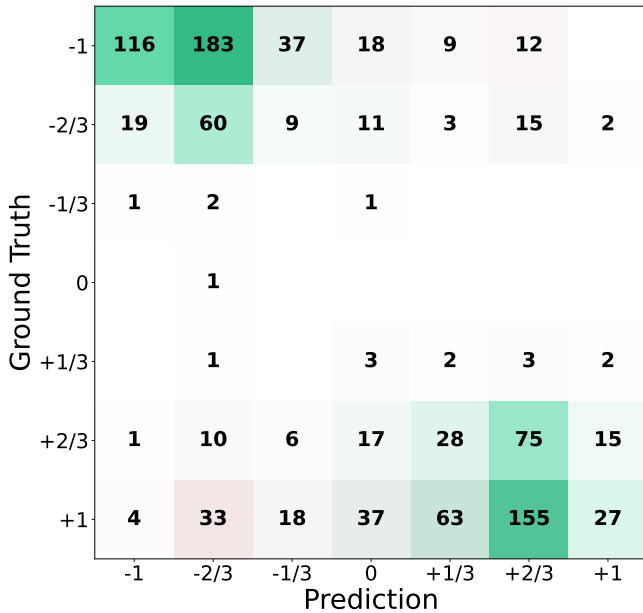
A.6. Confusion Matrices



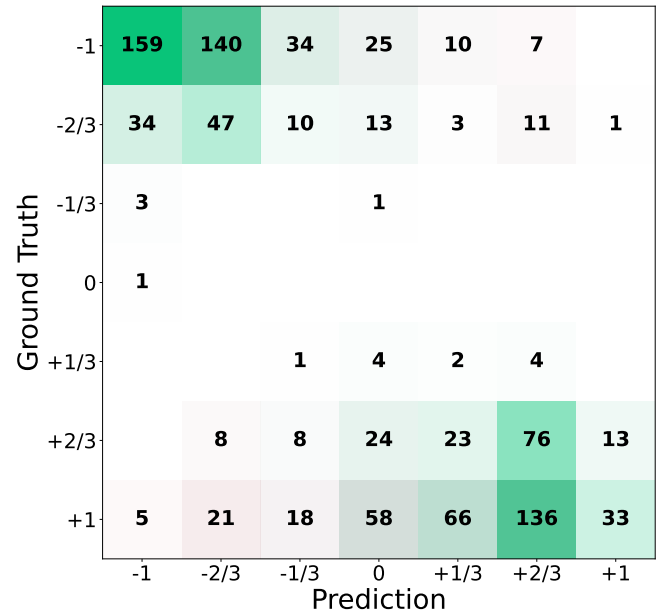
(a) GEMINI 3 FLASH Planner, CLAUDE OPUS 4.6 Judge



(b) GEMINI 3 FLASH Planner, GEMMA 4 31B Judge

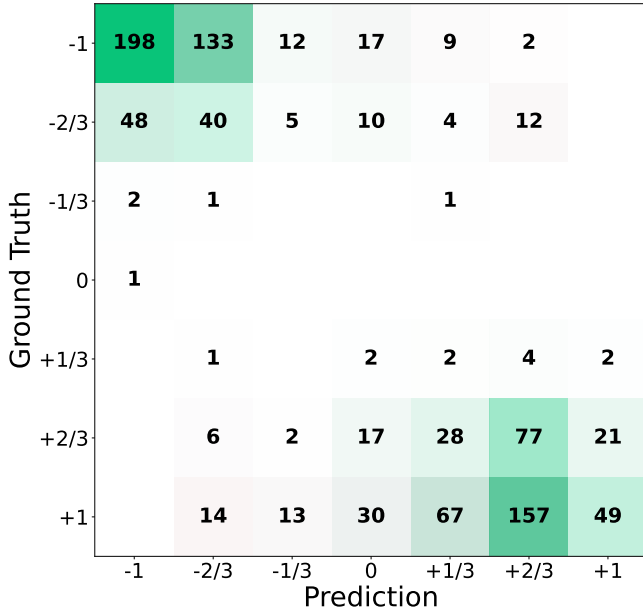


(c) GEMINI 3 FLASH Planner, QWEN 3.5 35B Judge

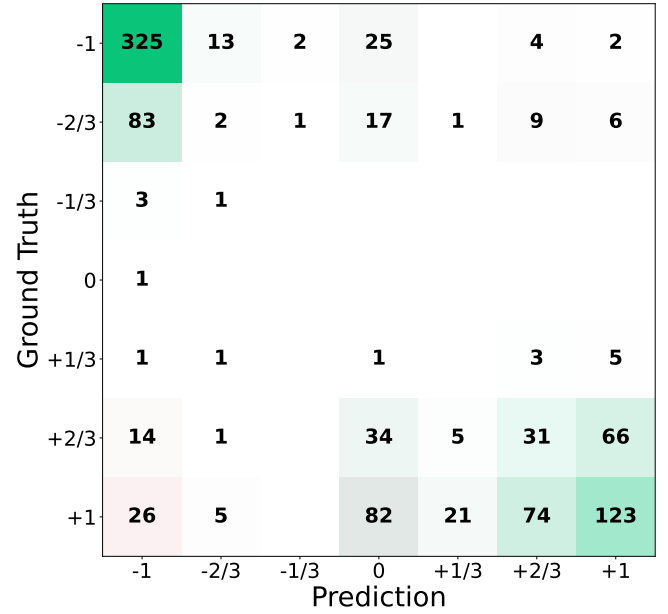


(d) GEMINI 3 FLASH Planner, QWEN 3.5 397B Judge

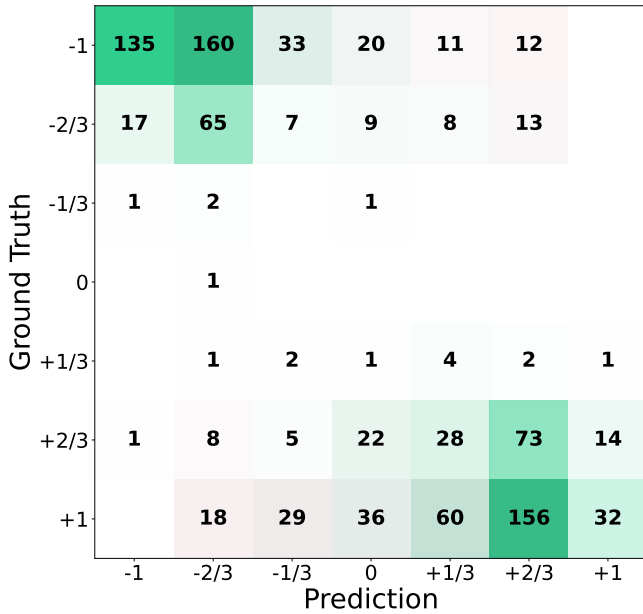
Figure A.2.: Confusion matrices for the rejudgets of the **Blueprint** run presented in Section 5.2. In all presented cases the **planner is Gemini 3 Flash**. The confusion matrix for the GEMINI 3 FLASH judge is in Figure 5.3.



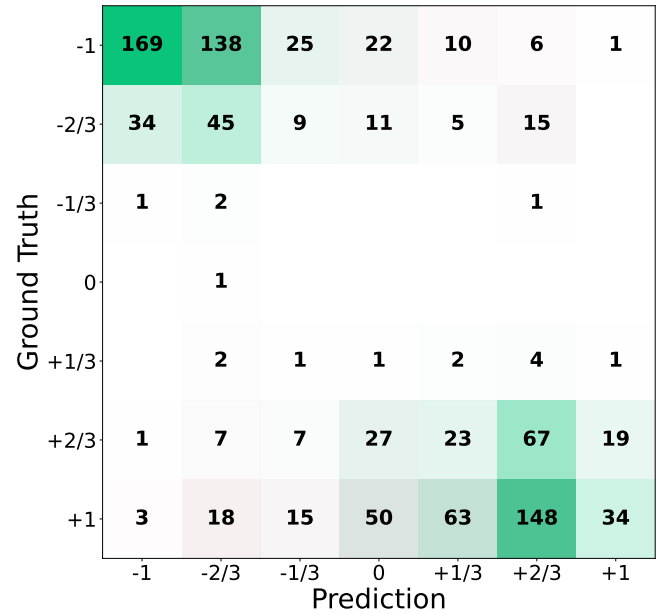
(a) GEMINI 3 FLASH Planner, CLAUDE OPUS 4.6 Judge



(b) GEMINI 3 FLASH Planner, GEMMA 4 31B Judge

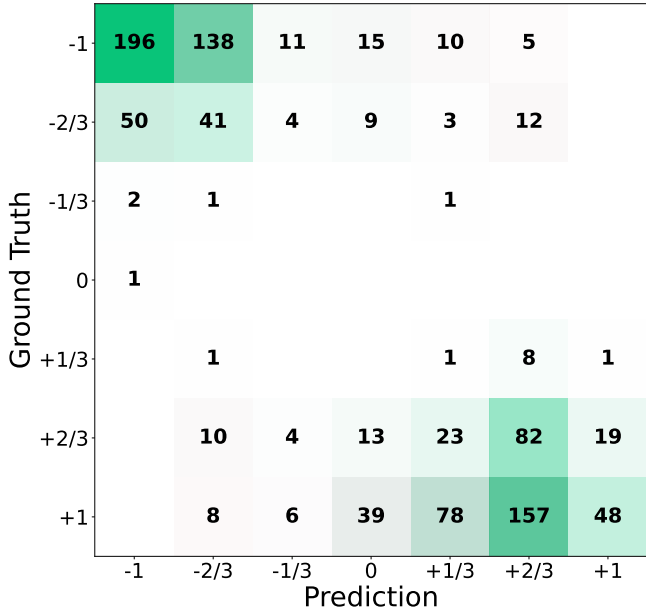


(c) GEMINI 3 FLASH Planner, QWEN 3.5 35B Judge

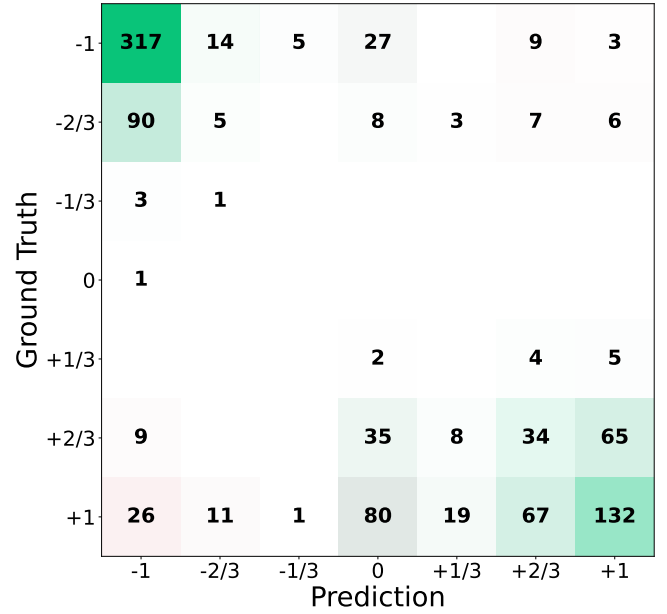


(d) GEMINI 3 FLASH Planner, QWEN 3.5 397B Judge

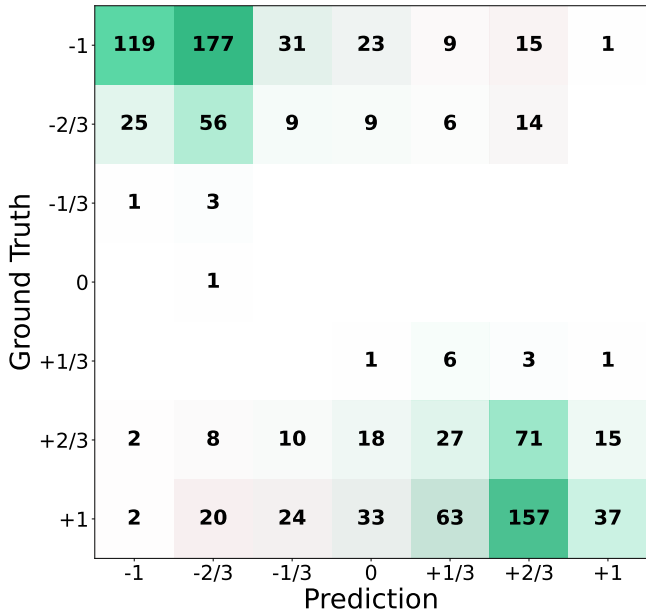
Figure A.3.: Confusion matrices for the rejudges of the **Empty-Strategy** run presented in Section 5.2. In all presented cases the **planner is Gemini 3 Flash**. The confusion matrix for the GEMINI 3 FLASH judge is in Figure 5.4.



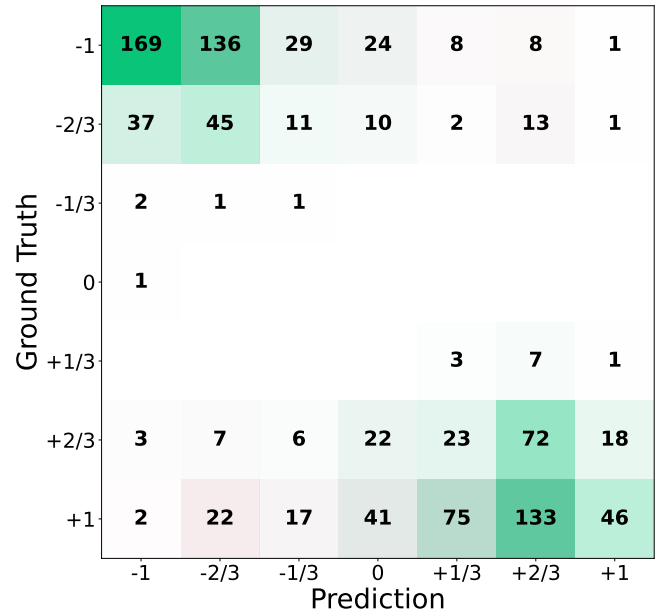
(a) GEMINI 3 FLASH Planner, CLAUDE OPUS 4.6 Judge



(b) GEMINI 3 FLASH Planner, GEMMA 4 31B Judge

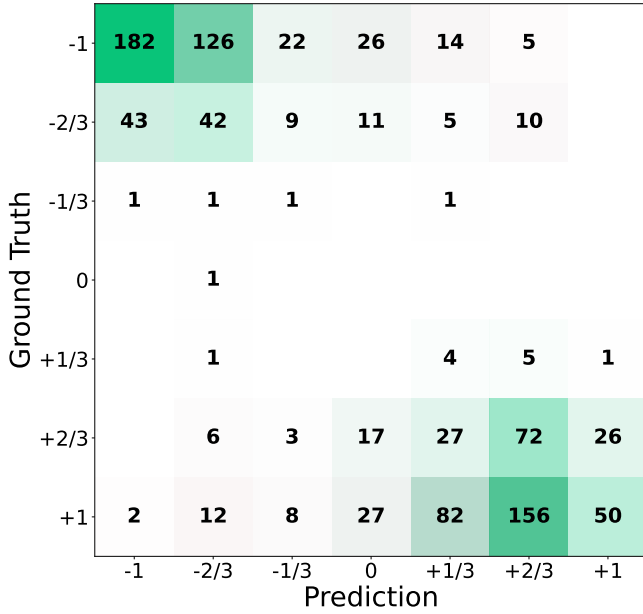


(c) GEMINI 3 FLASH Planner, QWEN 3.5 35B Judge

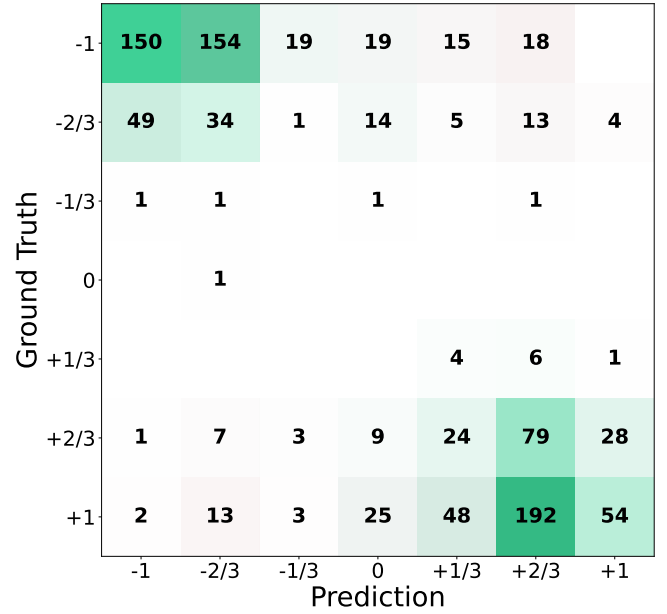


(d) GEMINI 3 FLASH Planner, QWEN 3.5 397B Judge

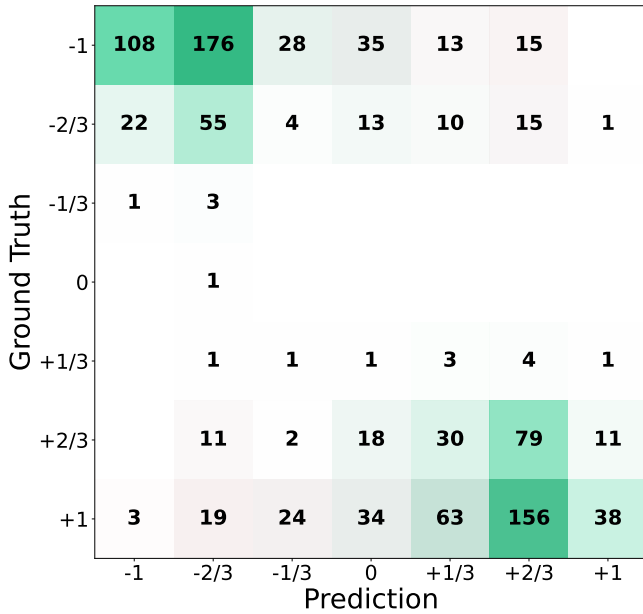
Figure A.4.: Confusion matrices for the rejudges of the **Single-Strategy** run presented in Section 5.2. In all presented cases the **planner is Gemini 3 Flash**. The confusion matrix for the GEMINI 3 FLASH judge is in Figure 5.4.



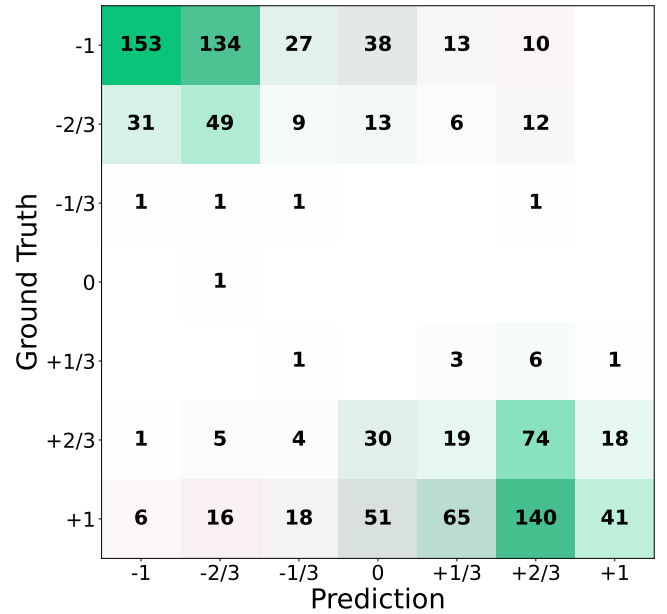
(a) GEMMA 4 31B Planner, CLAUDE OPUS 4.6 Judge



(b) GEMMA 4 31B Planner, GEMINI 3 FLASH Judge

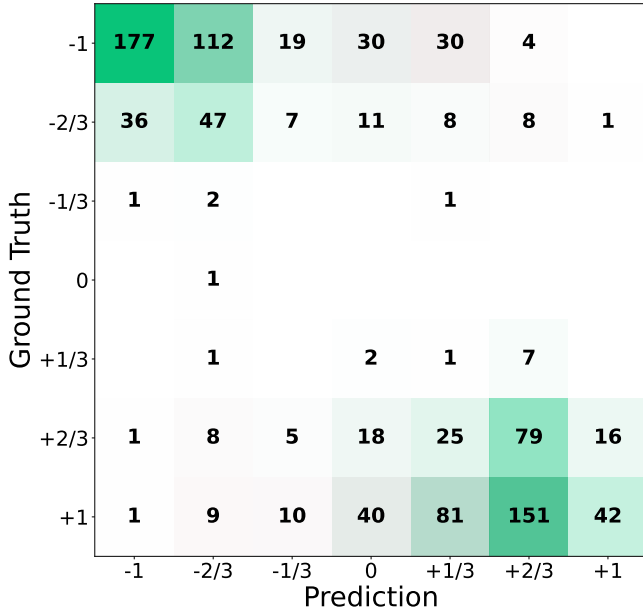


(c) GEMMA 4 31B Planner, QWEN 3.5 35B Judge

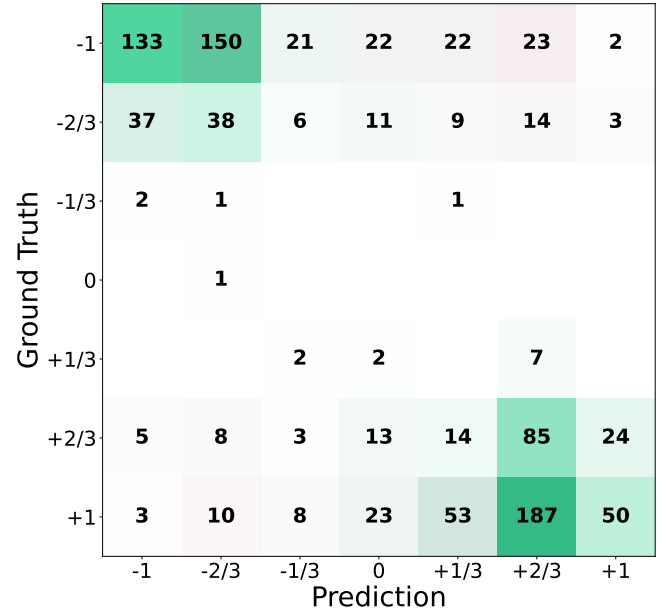


(d) GEMMA 4 31B Planner, QWEN 3.5 397B Judge

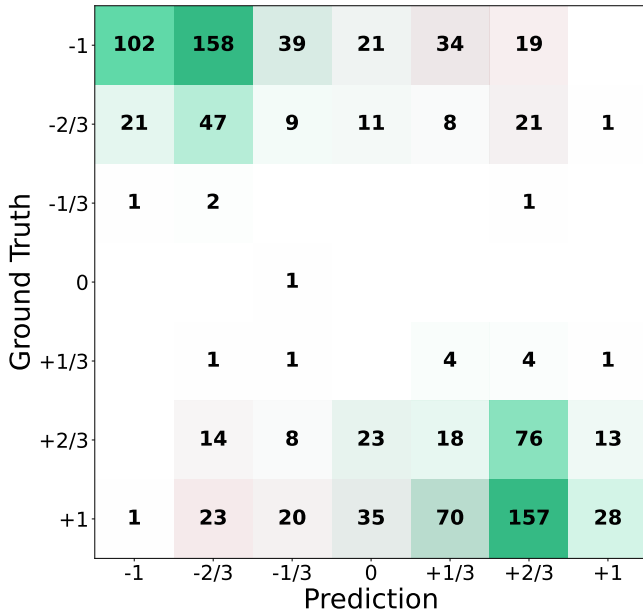
Figure A.5.: Confusion matrices for the rejudges of the **Blueprint** run presented in Section 5.2. In all presented cases the **planner is Gemma 4 31B**. The confusion matrix for the GEMMA 4 31B judge is in Figure 5.3.



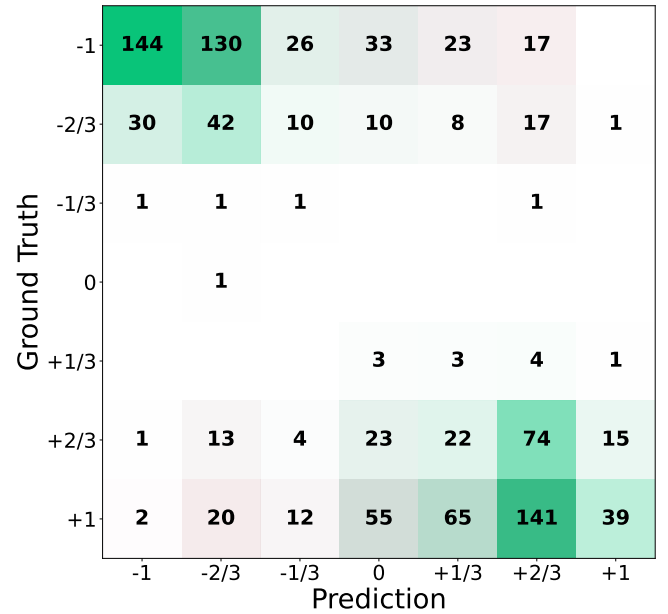
(a) GEMMA 4 31B Planner, CLAUDE OPUS 4.6 Judge



(b) GEMMA 4 31B Planner, GEMINI 3 FLASH Judge

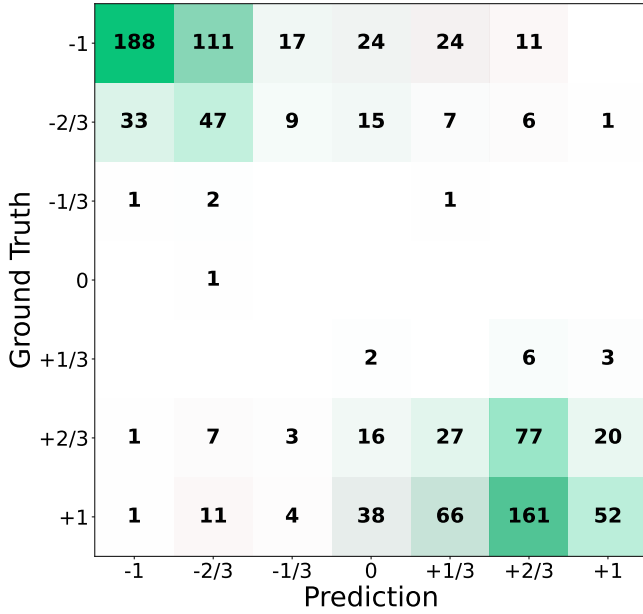


(c) GEMMA 4 31B Planner, QWEN 3.5 35B Judge

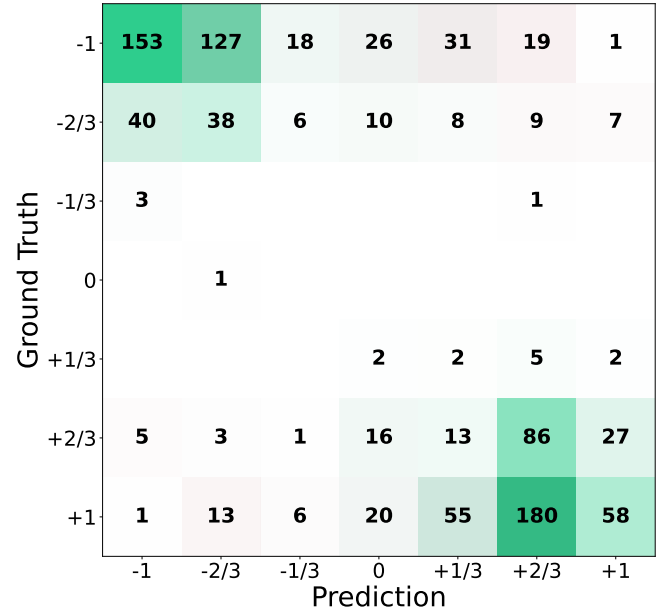


(d) GEMMA 4 31B Planner, QWEN 3.5 397B Judge

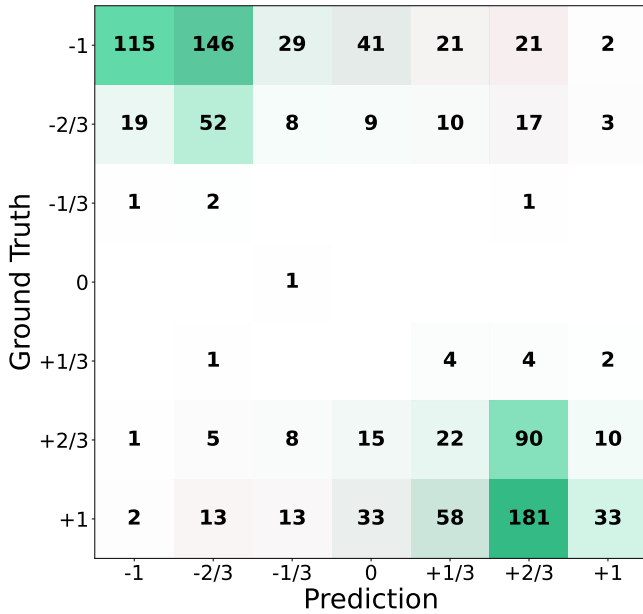
Figure A.6.: Confusion matrices for the rejudgets of the **Empty-Strategy** run presented in Section 5.2. In all presented cases the **planner is Gemma 4 31B**. The confusion matrix for the GEMMA 4 31B judge is in Figure 5.4.



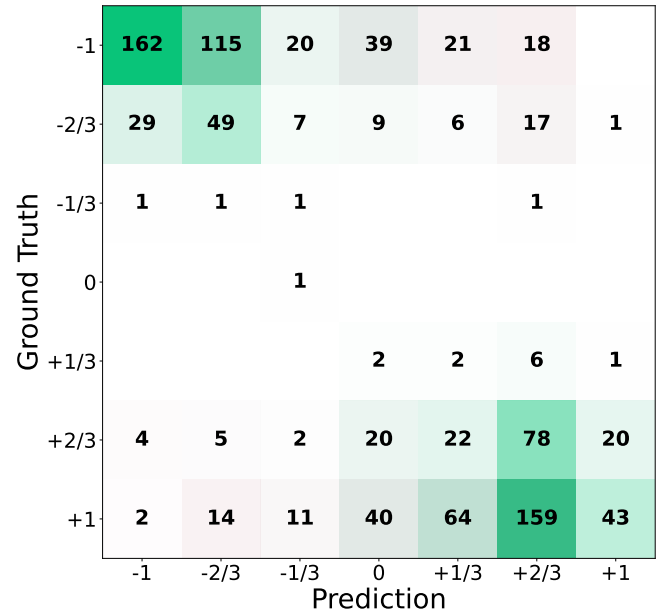
(a) GEMMA 4 31B Planner, CLAUDE OPUS 4.6 Judge



(b) GEMMA 4 31B Planner, GEMINI 3 FLASH Judge

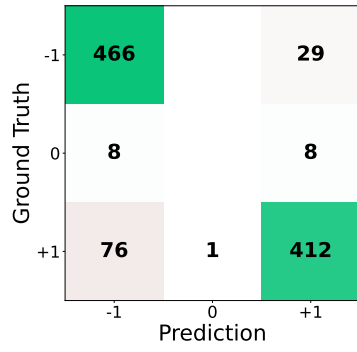


(c) GEMMA 4 31B Planner, QWEN 3.5 35B Judge

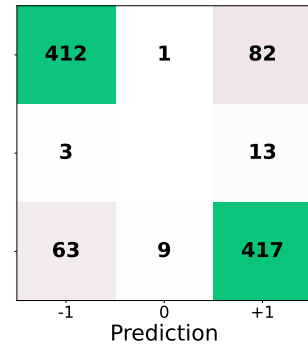


(d) GEMMA 4 31B Planner, QWEN 3.5 397B Judge

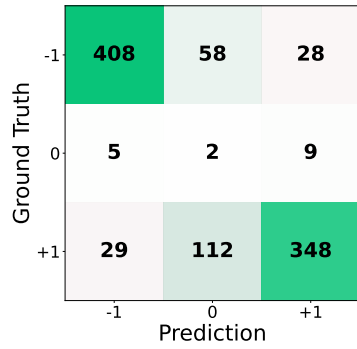
Figure A.7.: Confusion matrices for the rejudges of the **Single-Strategy** run presented in Section 5.2. In all presented cases the **planner is Gemma 4 31B**. The confusion matrix for the GEMMA 4 31B judge is in Figure 5.4.



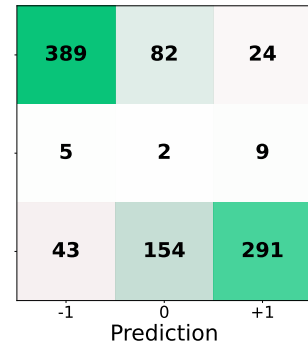
(a) GEMINI 3 FLASH Baseline



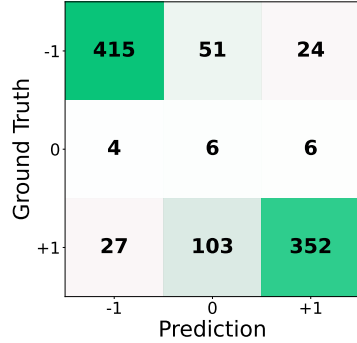
(b) GEMMA 4 31B Baseline



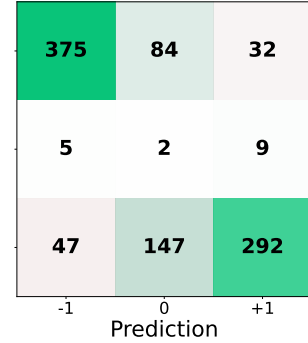
(c) GEMINI 3 FLASH Blueprints



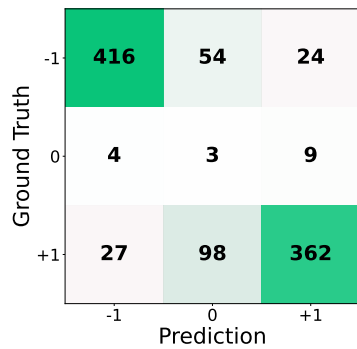
(d) GEMMA 4 31B Blueprints



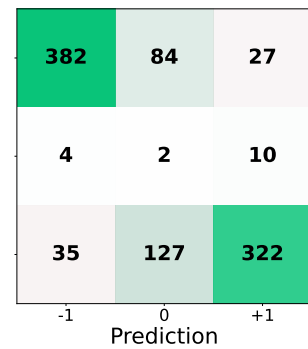
(e) GEMINI 3 FLASH Empty Strategy



(f) GEMMA 4 31B Empty Strategy



(g) GEMINI 3 FLASH Single Strategy



(h) GEMMA 4 31B Single Strategy

Figure A.8.: Confusion matrices for the full runs presented in Section 5.2 discretized to 3 classes. In all presented cases the planner is the same model as the judge.

A.7. The Learned Blueprint Pool

This appendix reproduces the learned blueprint pool in full, exactly as the pipeline loaded it for every run reported in Chapter 5. Section 5.1.3 describes how the pool was produced, and Section 4.1 defines the representation these files instantiate. Nothing here was authored by hand. Each file is the verbatim output of the synthesis and consolidation procedure of Section 4.4, run over the 2025 training quarters.

Every blueprint declares the same five top-level fields. `entry_conditions` and `selector_hints` carry the routing signal the LLM selector reads, `policy_constraints` bounds the actions and the iteration budget, `required_checks` names the questions a verdict must resolve, and `verification_graph` holds the nodes, their actions, and the transitions between them. Reading one `verification_graph` alongside Figure 5.2 is the quickest way to see what a cluster turned into.

The listings are ordered by the training traffic each blueprint absorbed, which is also roughly their order of influence on the results. Table A.7 indexes them. The generic blueprint carries no training traffic by construction, since it exists to absorb routing failures at inference time rather than to represent a cluster.

Blueprint	Training claims	Max. iterations	Listing
media_document_attribution_provenance_and_synthetic_origin	1983	9	A.1
primary_source_official_scientific_statistical_and_attribution_record_verification	1299	7	A.2
origin_and_corroboration_verifier	305	5	A.3
offer_authenticity	68	5	A.4
generic	0	4	A.5

Table A.7.: Index of the learned blueprint pool. Training claims count the clustered claims each blueprint was synthesized from, after consolidation merged fourteen clusters into four. The iteration budget is declared per blueprint and bounds how many planning rounds a claim routed to it may spend.

Listing A.1: `media_document_attribution_provenance_and_synthetic_origin`, synthesized from eight media-oriented clusters covering 1983 training claims. The largest blueprint in the pool and the destination of roughly 40% of evaluation traffic.

```
description: Verifies claims that TRAVEL WITH an attached image, video, screenshot,
official notice, magazine/press cover, poll graphic, digital card, advertisement,
or a video/audio clip of an attributed quote - establishing the media item's true
provenance, authorship, or synthetic origin via reverse-image/keyframe search, the
named entity's own channels, locating the original full recording, geolocation,
date/internal-consistency checks, and AI/deepfake detection where warranted, then
corroborating or contradicting its claimed context. A MATCH with the claimed context
is as valid an outcome as a MISMATCH. Distinct from
primary_source_official_scientific_statistical_and_attribution_record_verification
(which handles the same subject matter when it is a TEXT-only claim with no attached
media) and from origin_and_corroboration_verifier (which handles the narrower case
of a dated media event needing independent news corroboration but NO synthetic-origin,
quote-clip, or fabricated-document analysis).
entry_conditions:
  all:
    - feature: has_claim_text
      op: ==
      value: true
  any:
    - feature: asserts_recontextualization
      op: ==
```

```

    value: true
- feature: asserts_synthetic_origin
  op: ==
  value: true
- feature: asserts_place_or_date
  op: ==
  value: true
- feature: asserts_identity
  op: ==
  value: true
- feature: is_document_screenshot
  op: ==
  value: true
- feature: is_quote_attribution
  op: ==
  value: true
- feature: is_statistical
  op: ==
  value: true
- feature: is_scientific_medical
  op: ==
  value: true
- feature: has_image
  op: ==
  value: true
- feature: has_video
  op: ==
  value: true
name: media_document_attribution_provenance_and_synthetic_origin
policy_constraints:
  allowed_actions:
    - web_search
    - media
  max_iterations: 9
  require_counterevidence_search: true
required_checks:
- description: The item's true origin (earliest upload, original source, creator,
  original full broadcast, or - for documents/covers/notices/polls/ads/cards/screenshots
  - whether it exists at all and which account first posted it) was investigated
  via reverse-image search on keyframes or verbatim source/text lookup. A finding
  that the item matches its claimed context is as valid as a finding that it does
  not. Only an EXACT same-frame/same-image match counts as evidence about THIS media;
  PARTIAL/lookalike/stock-site/mirrored/other-event matches do not.
  id: media_origin_established
- description: When the claim attributes an item, statement, poll, cover, notice,
  digital card, advertisement, product, biography, statistic, law, budget, ruling,
  or policy to a named organisation, publication, official body, agency, court,
  or person, that entity's own official website / verified social channels / official
  records or registries were checked for whether the item genuinely originates from
  them, and where feasible the entity, creator, photographer, issuing authority,
  fact-check agency (e.g. PIB, TN Fact Check), or a relevant expert was consulted
  directly. Confirmation of authenticity and confirmation of fabrication are equally
  valid; report as unverified if no authoritative source can be reached.
  id: official_source_and_entity_check
- description: The claimed event, place, date, identity, authorship, or quote was
  compared against the item's established provenance using source lookup, official
  records, geolocation, date verification, or internal-consistency checks (e.g.
  wrong year, renamed streets, references to later-dated events, satire/parody/prank
  disclaimers). A MATCH and a MISMATCH are both valid outcomes.
  id: claimed_context_corroborated
- description: When the claim concerns what someone said or a specific edited/subtitled/dubbed/stitched
  clip, the original full-length recording was located (via watermark, official
  channel, or reverse-image search) and the relevant statements compared against
  the viral version at specific timestamps to detect stitching from different moments,
  mistranslation, dubbing, selective cropping, a verbal slip later corrected, an
  unrelated older speech relabelled, or AI-cloned audio laid over authentic video.
  Confirming the quote/context is faithful is as valid as finding it altered.

```

```

id: quote_or_edit_context_checked
- description: When the claim concerns AI-generation of image/audio/text, deepfakes/lip-sync manipulation, fabricated screenshots (wrong handle, misspelling, backdated publish date), or physical/astronomical/biological/technical/legal/calendar plausibility, synthetic-origin indicators, creator disclaimers/self-labels, tool watermarks (e.g. Veo, Gemini, Grok, SynthID, Midjourney, DALL-E), multiple AI-detection tools (noting they can be contradictory), and expert/official assessment were checked. For alleged statements by public figures, the authentic original recording was located and compared. Report as unverified if inconclusive.
id: synthetic_or_plausibility_checked
- description: When the claim asserts a figure, budget, law, policy, ruling, or official action attributed to a government, agency, court, or institution, the specific numbers/text/timeline were checked against primary official records (legislation text, official budget/statistics reports and their methodology manuals, court/parliament records, agency releases) and, where relevant, the methodology or comparison basis was verified against expert or official clarification. Confirming the claim is as valid as refuting it; report as unverified if primary records are inaccessible.
id: statistic_or_official_record_verified
- description: When the item may originate from a satirical, parody, prank, or meme source, or an impersonating/spoofed page or QR/link-scam, the originating account/website was traced and its self-description, 'About'/disclaimer text, and posting pattern were examined (e.g. self-declared satire sites such as Dunning-Kruger Times / America's Last Line of Defense, April-Fools captions, meme handles, impersonation domains with QR/link redirects). A genuine origin and a satirical/fabricated origin are equally valid findings; report as unverified if the origin cannot be traced.
id: satire_or_fabricated_source_checked
- description: A search for corrections, debunks, official fact-checks (e.g. PIB, TN Fact Check), the named entity's own denial/confirmation, or corroborating primary sources that could confirm or contradict the working conclusion was attempted. For a widely-covered alleged event, the complete absence of any credible corroborating report is itself a meaningful finding.
id: counterevidence_checked
selector_hints:
  negative:
    examples:
      - A text-only statistic, law, quote transcript, or scientific claim with no attached image/video - goes to primary_source_official_scientific_statistical_and_attribution_record_verification.
      - A dated media event claim needing only independent news corroboration, with no synthetic-origin, fabricated-document, or quote-clip question - goes to origin_and_corroboration_verifier.
      - A promotion/giveaway/subsidy/recruitment with a link or brand attribution needing domain resolution - goes to offer_authenticity.
    features: []
  positive:
    examples:
      - This attached viral image shows [named person] doing [action] / is [named person].
      - This attached clip/document/text is AI-generated, or this notice/magazine cover/poll/ad/digital card image is authentic.
      - This screenshot claims [named organisation/publication/official body] issued this statement/graphic/cover/poll/notice.
      - This attached clip shows [official/public figure] saying [quote] (video/audio clip must be authenticated, not a text-only quote).
      - This satire/parody article image or fabricated screenshot is being shared as a genuine announcement.
      - This graphic/screenshot presents a government statistic, budget, law, or ruling and the image itself must be traced to its originating account.
    features:
      - has_image
      - has_video
      - is_multimodal
      - asserts_synthetic_origin
      - asserts_identity
      - is_document_screenshot
verification_graph:
  nodes:
    - actions:
        - action: media
          intent: 'Establish the item's true origin: run reverse-image search on multiple

```

keyframes/frames to find the earliest upload and original source. Record whether each result is an EXACT match (same frames/image) or only PARTIAL/similar. Compare the true origin with the claimed context - a MATCH is as valid a finding as a MISMATCH.'

query_guidance: Extract several keyframes and reverse-image search each (Google Lens, TinEye, InVID-WeVerify, Bing). Treat only EXACT same-image/same-frame matches as evidence about THIS media; PARTIAL matches, stock-site lookalikes, mirrored/flipped copies, and results for OTHER media of the same event are not evidence. Note original upload date, uploader, watermark, on-screen logo, and any AI-tool watermark (Veo, Gemini, Grok, SynthID, Midjourney, DALL-E). For documents/screenshots/covers/polls/ads/cards, also search the exact text/headline/handle verbatim to find its earliest source or existence, and note misspellings, wrong usernames, or backdated publication dates that reveal fabrication.

- action: web_search

intent: 'Establish the core factual context: search for credible reports of the specific claimed event/place/date/identity/statement/poll/statistic/fact to locate likely primary sources and any existing fact-checks. Absence of any credible report and confirmation by credible reports are both valid findings.'

query_guidance: Search the claimed actors, location, date, and quoted text in the local language and English. Look for official statements, news coverage, government fact-checks (e.g. PIB, TN Fact Check), and prior debunks referencing THIS item. Note that for a widely-covered event, the complete absence of corroborating reports is itself meaningful.

activates_checks:

- media_origin_established
- claimed_context_corroborated

id: layer0_origin

transition:

- if: continue
- to: layer1_synthesis

type: actions

- activates_checks: []

id: layer1_synthesis

transition:

- if: the claim concerns AI-generation of image/audio/text, deepfake/lip-sync editing, a fabricated screenshot/notice, or physical/astronomical/biological/technical/legal/calendar plausibility, or the item shows synthetic artifacts or an AI-tool watermark that needs assessment
- to: layer2_synthetic
- if: the claim asserts a figure, budget, law, ruling, or official policy/action attributed to a government, agency, court, or institution whose numbers or text must be checked against primary official records
- to: layer2_official_source
- if: the claim attributes an item, statement, poll, cover, notice, digital card, advertisement, product, biography, or policy to a named organisation, publication, official body, agency, or person whose authenticity or existence must be confirmed against that entity's own channels/registries, a fact-check agency, or by contacting it; or the item may originate from a satire/parody/meme/impersonating/QR-scam account whose nature must be established
- to: layer2_official_source
- if: the claim concerns what someone said in a video/audio clip and the original full recording must be located to check for stitching, mistranslation, dubbing, selective editing, a corrected verbal slip, an unrelated older speech relabelled, or AI-cloned audio over real video
- to: layer2_quote_context
- if: the item appears authentic but the claimed event/place/date/identity may be wrong, and provenance points to a geolocatable scene, specific creator, or datable original upload that needs direct confirmation
- to: layer2_primary_source
- if: origin and claimed context are already established with sufficient EXACT-match evidence in either direction
- to: finalize
- if: evidence is insufficient, contradictory, or the true origin remains unresolved
- to: layer2_primary_source

type: synthesis

- actions:

- action: media

intent: 'Assess whether the media/audio/text is AI-generated or manipulated, whether a screenshot/notice was fabricated, or whether the underlying scientific/technical/biological/legal/calendar

claim is plausible: look for generation artifacts, deepfake/lip-sync inconsistencies, creator disclaimers or AI self-labels on the original post, AI-tool watermarks, spelling errors / wrong handles / backdated publish dates in screenshots, and run/consult multiple AI-detection tools (image, audio, and text where relevant). If the claim alleges a specific statement by a public figure, ALSO locate the authentic original recording and compare it directly, and consider whether the audio may be AI-cloned even when the video is genuine. Confirming authenticity/plausibility is as valid as detecting synthesis/implausibility; note AI detectors can be contradictory.'

query_guidance: Inspect for visual/audio inconsistencies (unnatural lip-sync, distorted hands/fingers, glitching logos/nameplates, gibberish/blurred text, morphing backgrounds, reused audio, missing microphones, lack of camera shake). Check the original upload's caption/bio/comments for disclaimers or self-labels ('AI generated', '#ai', 'scripted drama', 'parody/satire', 'April fools') and for tool watermarks (Veo, Gemini, Grok, SynthID, Midjourney, DALL-E). For fabricated screenshots, verify the exact username, spelling, and publication/backdate against the genuine account. Cross-check physical/astronomical/biological/technical/legal/calendar plausibility against authoritative references (official product specs, endangered-species lists, legislation text, geological/astronomical/seismic data, ingredient/additive lists, calendar math). For alleged official statements, find the genuine full recording (ANI, Newsweek, ISPR, official livestream) and compare timestamps and audio. Report as unverified if inconclusive.

- action: web_search

intent: Corroborate the plausibility/synthetic/fabrication assessment with expert commentary, official records, or existing fact-checks on this specific item.

query_guidance: Search for expert statements (deepfake analysis units such as DAU, entomologists, geologists, astronomers, physicists, publishing/AI-detection experts, constitutional/legal scholars, product manufacturers), official data (satellite/seismic records, official denials, legislation, PIB/agency fact-checks), and prior debunks or confirmations referencing THIS item.

activates_checks:

- synthetic_or_plausibility_checked
- satire_or_fabricated_source_checked

id: layer2_synthetic

transition:

- if: continue
- to: layer3_corroboration

type: actions

- actions:

- action: web_search

intent: 'Confirm whether the item, figure, or action genuinely originates from the named entity or exists at all: check that entity's own official website, verified social accounts, and official records/registries for the claimed cover, statement, poll, notice, digital card, advertisement, product, biography, statistic, budget, law, ruling, or policy. Also determine whether the item traces to a satirical, parody, prank, meme, impersonating, or QR/link-scam source. Confirmation of authenticity and confirmation of fabrication/satire are equally valid.'

query_guidance: Search the official domain and verified handles of the named organisation/publication/agency/person for the exact item or headline. Check official registries or records (municipal records, parliament/legislation sites, court records, official budget/statistics reports and their methodology manuals, endangered-species lists, product/publisher catalogues, poll-firm archives, magazine issue archives, cardinal/office registries, development-agency project lists). For statistical/policy claims, verify the specific numbers, text, timeframe, and comparison basis against the primary official source and note any methodological caveats. If the item may be satire/parody/prank, trace the originating account/website and read its 'About'/disclaimer text and posting pattern (self-declared satire sites such as Dunning-Kruger Times / America's Last Line of Defense, April-Fools captions, meme handles). Where the item is an advertisement, notice, or card with an embedded link or QR code, check where that link/QR redirects (impersonation, phishing/scam, closed channel). Also check relevant fact-check agencies (PIB, TN Fact Check) for official clarifications. Note whether the entity's own sources carry the item,

deny it, or are silent; a fabricated issue-number/cover/poll often does not exist on the official archive at all.

- action: web_search
 - intent: Where feasible, obtain direct confirmation from the entity, creator, photographer, issuing authority, agency spokesperson, or a relevant domain expert. A statement confirming authenticity/accuracy is as valid as one confirming fabrication; report as unverified if no authoritative response is available.
 - query_guidance: Look for direct verifications reported by fact-checkers (org spokespersons, editors, CEOs, creators, embassy press officers, ministry/agency press offices, statisticians, experts, service agencies) or the entity's own disclaimer posts. Note any explicit confirmation or denial and its source. Check for internal inconsistencies (outdated street names, wrong dates, non-existent issue numbers, mismatched timelines, satire/parody/prank disclaimers) that reveal fabrication or misattribution.
- activates_checks:
 - official_source_and_entity_check
 - statistic_or_official_record_verified
 - satire_or_fabricated_source_checked
 - claimed_context_corroborated
- id: layer2_official_source
- transition:
 - if: continue
 - to: layer3_corroboration
- type: actions
- actions:
 - action: web_search
 - intent: Locate the original, full-length recording of the speaker/event and compare the claimed statement against what was actually said at specific timestamps. Confirming the quote is faithful is as valid as finding it stitched, mistranslated, dubbed, cropped out of context, a verbal slip later corrected, an unrelated older speech relabelled, or fitted with AI-cloned audio.
 - query_guidance: Trace watermarks and reverse-image results to the original broadcaster's channel (news agency YouTube, official livestream, parliament/press-secretariat channel). Compare specific timestamps; note if sentences were stitched from different moments, subtitles/dubbing differ from the original-language audio, context was removed, the speaker corrected a slip, or the clip is an older speech on an unrelated topic. Also check whether the audio may be AI-cloned even when the video is real (run audio through AI-voice detectors).
 - action: web_search
 - intent: Corroborate the quote finding with official clarifications, the speaker's own statements, or prior fact-checks.
 - query_guidance: Search official fact-check bodies (e.g. PIB, TN Fact Check), the speaker's verified accounts, party/organisation spokespersons, and independent coverage that confirms or contradicts the attributed quote.
- activates_checks:
 - quote_or_edit_context_checked
 - claimed_context_corroborated
- id: layer2_quote_context
- transition:
 - if: continue
 - to: layer3_corroboration
- type: actions
- actions:
 - action: web_search
 - intent: 'Confirm the claimed source, date, or location directly against a primary source: the official website/social channels of any named body, the original creator/photographer, the issuing authority, or geolocation of the scene. Establish the earliest upload date to test date/recontextualization claims. Confirmation and refutation are equally valid.'
 - query_guidance: Check the official website or verified accounts of any named organization/publication/agency; trace the original uploader or photographer; geolocate landmarks, signs, licence plates, and buildings against Street View/maps; verify signatory or subject identities. Compare the item's earliest confirmed date/place with the claimed one - a recirculated real video is often revealed by an earlier upload or a later-dated event it supposedly precedes.
 - action: web_search
 - intent: 'Search explicitly for counter-evidence and corroboration: corrections,

```

    debunks, official fact-checks, the named entity's own denial/confirmation,
    or independent reports that could confirm or contradict the working conclusion.'
  query_guidance: Look for fact-checks of THIS item, official denials/confirmations,
    and independent coverage that either matches or breaks the claimed context.
    For a widely-covered alleged event, note that the complete absence of any
    credible report is itself a meaningful finding.
  activates_checks:
  - claimed_context_corroborated
  - counterevidence_checked
  id: layer2_primary_source
  transition:
  - if: continue
    to: layer3_corroboration
  type: actions
- actions:
  - action: web_search
    intent: Corroborate the leading finding with at least one additional independent
      source, resolving any remaining discrepancy between the item's established
      origin/existence and its claimed context.
    query_guidance: Seek a second independent confirmation (different outlet, official
      record, or expert). Confirm that any 'original' or debunk cited is an EXACT
      match to THIS media, not a similar or partial one. If the true origin cannot
      be established with an EXACT match, or the item's existence/figure cannot
      be confirmed or denied by an authoritative source, report the provenance as
      unverified rather than assuming a false context.
  activates_checks:
  - media_origin_established
  - counterevidence_checked
  id: layer3_corroboration
  transition:
  - if: continue
    to: layer3_synthesis
  type: actions
- activates_checks: []
  id: layer3_synthesis
  transition:
  - if: continue
    to: finalize
  type: synthesis
start_node: layer0_origin

```

Listing A.2: primary_source_official_scientific_statistical_and_attribution_record_verification, synthesized from four record-oriented clusters covering 1299 training claims. It handles textual assertions that resolve against an authoritative record.

```

description: Verifies TEXT-ONLY factual claims that carry no attached image or video
  - statistics, laws/decrees/rulings/official actions, event assertions, scientific/medical
    claims, and quote/attribution claims - by tracing them to their primary source (official
    gazettes, registries, legislatures, courts, ombudsmen, statistical agencies, the
    named institution/spokesperson/author, peer-reviewed literature, or health authorities)
    and cross-checking the definitive authoritative record. Confirmation and refutation
    are equally valid. Distinct from media_document_attribution_provenance_and_synthetic_origin
    in that the claim arrives as text/document reference and the decisive evidence is
    documentary/official/scientific/statistical or the recovered original of an attributed
    statement, NOT an attached image/video requiring visual forensics.
entry_conditions:
  all: []
  any:
  - feature: is_statistical
    op: ==
    value: true
  - feature: is_document_screenshot
    op: ==
    value: true
  - feature: is_quote_attribution
    op: ==

```

```

    value: true
  - feature: is_scientific_medical
    op: ==
    value: true
  - feature: asserts_place_or_date
    op: ==
    value: true
  - feature: asserts_recontextualization
    op: ==
    value: true
name: primary_source_official_scientific_statistical_and_attribution_record_verification
policy_constraints:
  allowed_actions:
    - web_search
    - media
  max_iterations: 7
  require_counterevidence_search: true
required_checks:
- description: The claim was traced to its earliest/original source (post, article,
  video, broadcast, press briefing, cited study, screenshot, or official document)
  to establish exactly what was asserted, by whom, and when.
  id: primary_source_traced
- description: 'The origin and credibility of the source carrying the claim were assessed:
  is it an official body, peer-reviewed outlet, or reputable news organisation,
  versus an anonymous account, a newly-registered or impersonating domain, a satirical
  site, a known propaganda/disinformation outlet, an edited screenshot, a spliced/mislabelled
  clip, or AI-generated media? Establishing that the source is legitimate is as
  valid a finding as establishing that it is fabricated; where provenance cannot
  be determined, report it as unverified rather than assuming fabrication.'
  id: source_provenance_and_legitimacy
- description: 'The relevant definitive record was consulted and compared against
  the claim - for legal/statistical/event/attribution claims: official gazette/registry,
  legislative or decree text, government/court/ombudsman database, statistical agency,
  or the named institution directly; for scientific/medical claims: peer-reviewed
  literature and the guidance/databases of authoritative health bodies (WHO, FDA,
  EMA, national health agencies, cancer institutes). A MATCH (claim confirmed or
  partially confirmed) is as valid a finding as a MISMATCH; where the record is
  genuinely silent, report that rather than assuming the claim is false.'
  id: authoritative_record_checked
- description: For scientific, medical, technical, or specialist-legal claims, qualified
  experts or the documented scientific/clinical/legal consensus were consulted to
  assess plausibility and current evidence. Confirming that the claim aligns with
  consensus is an equally acceptable outcome; where evidence is genuinely absent,
  preliminary, or unsettled, report it as such rather than assuming the claim is
  false.
  id: expert_or_scientific_consensus_checked
- description: Any attributed statement, figure, cited study, statistic, or document
  was checked against its FULL original context - the complete video/transcript/paper/document,
  not the clipped excerpt - to confirm it was not altered, mistranslated, spliced,
  misdated, satirically fabricated, or stripped of qualifying context. Confirming
  that the quote is accurate and in-context is as valid a finding as showing it
  was distorted. For statistics, verify the exact definition, denominator, unit,
  time period, and methodology behind the figure (e.g. cases vs unique patients,
  households vs individuals, per-capita vs total, index choice, seasonally-adjusted
  vs annual, an in-vitro/preclinical result presented as a proven human cure, or
  a draft presented as a signed law).
  id: quote_or_context_fidelity
- description: Independent corroboration and possible contradictions (corrections,
  fact-checks, competing official data, replication or rebuttal studies, denials
  or confirmations from named parties) were sought before finalising.
  id: corroboration_or_counterevidence
selector_hints:
  negative:
    examples:
      - This viral photo/video/screenshot/cover/notice shows or says X - goes to
        media_document_attribution_provenance_and_synthetic_origin
        because an attached image/video must be authenticated.

```

```

- This viral image/clip is AI-generated - goes to media_document_attribution_provenance_and_synthetic_origin
  (synthetic-media forensics).
- A dated video claimed to show a specific event at a named place needing independent
  news corroboration - goes to origin_and_corroboration_verifier.
features:
- asserts_synthetic_origin
- has_image
- has_video
positive:
examples:
- A government signed a decree/law/executive order requiring X (text claim, verifiable
  against the official registry or gazette, no image attached).
- Official records show politician Y attended Z of N legislative sessions (text-only
  attendance/vote figures).
- Text claim that an institution/parliament/ombudsman/court ruled or stated X
  about person or country Y.
- 'Text statistic: registered divorces/GDP/employment/spending changed by X between
  two years or reached value N (no accompanying graphic).'


---



```

```

- source_provenance_and_legitimacy
id: layer0_scope
transition:
- if: continue
  to: layer1_synthesis
type: actions
- activates_checks: []
id: layer1_synthesis
transition:
- if: the primary source, original media, and authoritative records already confirm
  or refute the claim with sufficient, uncontradicted evidence
  to: finalize
- if: the definitive official record, scientific literature, statistical dataset,
  full original quote/transcript, or expert assessment still needs to be consulted,
  or evidence is thin, contradictory, or context-dependent
  to: layer2_records
type: synthesis
- actions:
- action: web_search
  intent: Consult the authoritative record directly and compare it point-by-point
  with the claim's specific assertions, figures, or wording - confirming or
  partially confirming the claim is an equally acceptable outcome. For scientific/medical
  claims, review peer-reviewed studies and the published position of relevant
  health authorities. For statistics, retrieve the exact dataset and verify
  the definition behind the figure. For attributed statements, retrieve the
  full original speech/transcript/article and read the surrounding context.
  query_guidance: 'Go to the definitive primary source: legislative/decreed text
  and its number, government or ministry database, court/ombudsman ruling, official
  statistics portal (e.g. national statistics office, Eurostat, USDA, BLS, OBR,
  CBO, central bank), the institution's own published statement, the full original
  video/transcript of a speech or interview, OR for scientific/medical claims
  the relevant peer-reviewed literature (PubMed, journals) and authoritative
  health-body guidance. Retrieve exact figures, dates, wording, denominators,
  methodology, and study conclusions. Where the claim quotes someone or cites
  a study, retrieve the full original statement/transcript/paper and check whether
  it was omitted, altered, mistranslated, misdated, satirically fabricated,
  or misrepresented (e.g. preclinical/in-vitro findings framed as proven human
  results, cases counted as unique patients, households counted as individuals,
  or a draft framed as a signed law). An accurate, in-context match confirms
  the claim.'
- action: web_search
  intent: 'Seek independent corroboration and any contradicting evidence: other
  credible reporting, competing official data, qualified expert opinion, replication
  or rebuttal studies, prior fact-checks, denials or confirmations from named
  parties, or corrections.'
  query_guidance: Search for other reputable outlets, official confirmations,
  or expert/clinician commentary, and separately for corrections, denials, fact-checks,
  or competing studies/datasets that either corroborate or contradict the evidence
  so far. Weigh source credibility (official/institutional/peer-reviewed vs
  anonymous/satirical/propaganda/unqualified). If the record cannot establish
  the fact either way, plan to report it as unverified rather than defaulting
  to a verdict.
activates_checks:
- authoritative_record_checked
- expert_or_scientific_consensus_checked
- quote_or_context_fidelity
- corroboration_or_counterevidence
id: layer2_records
transition:
- if: continue
  to: layer2_synthesis
type: actions
- activates_checks: []
id: layer2_synthesis
transition:
- if: the public record and literature already resolve the claim, or direct contact
  with the named institution/author/spokesperson is not needed

```

```

    to: finalize
  - if: the public record is silent, ambiguous, or contradictory and the named institution,
      spokesperson, study author, claimant, or the person to whom a statement is
      attributed should be contacted directly to confirm the exact figure, wording,
      source, or interpretation
    to: layer3_direct_outreach
  type: synthesis
- actions:
  - action: web_search
    intent: Obtain a direct statement or the underlying source from the named institution,
      statistical agency, study author, spokesperson, press office, or the claimant/attribution
      person themselves to resolve a figure, quote, legal interpretation, or attribution
      that the public record left ambiguous. A response confirming the claim is
      as valid an outcome as one refuting it.
    query_guidance: Identify and reach the relevant press office, ministry, agency,
      expert, publisher, or the person/organisation to whom a statement or policy
      is attributed, and request the specific dataset, decree text, methodology,
      transcript, or clarification at issue (e.g. the source of a spokesperson's
      statistic, whether a figure counts cases or patients, whether a named study
      says what is claimed, or whether an attributed quote was actually said). Note
      explicitly if the party does not respond or cannot substantiate the claim.
      If the fact still cannot be established, report it as unverified rather than
      assuming a verdict.
  activates_checks:
  - expert_or_scientific_consensus_checked
  - quote_or_context_fidelity
  - corroboration_or_counterevidence
  id: layer3_direct_outreach
  transition:
  - if: continue
    to: layer3_synthesis
  type: actions
- activates_checks: []
  id: layer3_synthesis
  transition:
  - if: continue
    to: finalize
  type: synthesis
start_node: layer0_scope

```

Listing A.3: origin_and_corroboration_verifier, one of the two clusters that survived consolidation unmerged, covering 305 training claims.

```

description: Verifies dated media-EVENT claims - an image or video asserted to show
  a specific event at a named place and date - by locating the original via reverse-image/keyframe
  search and corroborating it against an independent authoritative source (news reports
  or official bodies). Handles only the corroboration path; does NOT perform synthetic-origin/AI
  detection, quote-clip authentication, or fabricated-document/cover analysis. Distinct
  from media_document_attribution_provenance_and_synthetic_origin, which takes any
  media claim requiring authenticity forensics, impersonation/fabrication tracing,
  or synthetic-origin detection; this blueprint is the narrow residual where the sole
  question is whether a genuine dated media event happened as described.
entry_conditions:
  all: []
  any:
  - feature: asserts_recontextualization
    op: ==
    value: true
  - feature: asserts_place_or_date
    op: ==
    value: true
  - feature: is_statistical
    op: ==
    value: true
  - feature: is_scientific_medical
    op: ==

```

```

    value: true
  - feature: is_document_screenshot
    op: ==
    value: true
name: origin_and_corroboration_verifier
policy_constraints:
  allowed_actions:
  - web_search
  - media
  max_iterations: 5
  require_counterevidence_search: true
required_checks:
- description: The original/primary source of the media or data was sought (earliest
  upload, original document, or authoritative dataset). Report as unverified if
  the origin cannot be established.
  id: primary_source_located
- description: The claimed place, date, and context were compared against the established
  original context. A MATCH confirming the claim is as valid a finding as a MISMATCH.
  id: context_and_date_verified
- description: For media claims, any reverse-image/fact-check match was assessed as
  EXACT (same frames/image = evidence about THIS media) versus PARTIAL/lookalike/other-media-of-same-event
  (NOT evidence about this media).
  id: media_referent_discipline
- description: The finding was corroborated with an independent authoritative source
  (news report, official body, or domain expert), or the gap was reported as unverified.
  id: corroboration_obtained
- description: Counterevidence or contradicting sources were sought before finalizing.
  id: counterevidence_checked
selector_hints:
  negative:
    examples:
    - This image/video was generated by AI or is a deepfake - goes to
      media_document_attribution_provenance_and_synthetic_origin.
    - Did person X say quote Y in this clip? - goes to media_document_attribution_provenance_and_synthetic_origin
      (quote-clip authentication).
    - A fabricated screenshot/cover/notice must be traced to its originating account
      - goes to media_document_attribution_provenance_and_synthetic_origin.
    - A text-only statistic or scientific claim with no attached media - goes to
      primary_source_official_scientific_statistical_and_attribution_record_verification.
  features:
  - asserts_synthetic_origin
  - is_quote_attribution
  positive:
    examples:
    - A viral video is claimed to show a specific real event at a named place on a
      named date, and the question is simply whether that event occurred as shown.
    - A dated photo of an incident needs independent news corroboration to confirm
      it depicts the stated occurrence.
  features:
  - has_image
  - has_video
  - is_multimodal
  - has_date
verification_graph:
  nodes:
  - actions:
  - action: media
    intent: Establish the true origin and original context of any attached media,
    then compare it with the claimed context. A MATCH is as valid a finding as
    a MISMATCH.
    query_guidance: Extract keyframes and run reverse-image/video search to find
    the EARLIEST online occurrence. Treat a match as evidence about THIS media
    only if it is EXACT (same frames/image); PARTIAL matches, stock lookalikes,
    and debunks of other media of the same event are not evidence about this media.
    If origin cannot be established, note it as unverified rather than assuming
    any conclusion.
  - action: web_search

```

```

intent: Establish the core factual context and locate the primary/authoritative
source for the claim (original document, official dataset, or first credible
report).
query_guidance: Search for the named event, place, date, statistic, or entity.
For data/scientific claims, go to primary sources (official agencies, published
studies, original documents). For event/media claims, look for the earliest
news coverage describing the same event.
activates_checks:
- primary_source_located
- media_referent_discipline
id: layer0_origin
transition:
- if: continue
to: layer1_assess
type: actions
- activates_checks:
- context_and_date_verified
id: layer1_assess
transition:
- if: the primary source and context are clearly established and the claim is
already confirmed or refuted with sufficient corroboration
to: finalize
- if: the origin/context is established but needs an independent corroborating
source, or evidence is insufficient or contradictory
to: layer2_corroborate
type: synthesis
- actions:
- action: web_search
intent: Corroborate the established origin/context with an independent authoritative
source and actively seek any contradicting evidence.
query_guidance: 'Cross-check with a second independent authority: official bodies,
published records/datasets, domain experts, or a local reporter for on-ground
events. Search explicitly for corrections, fact-checks, or sources that contradict
the finding so far. Confirming the claim is an equally acceptable outcome.'
- action: media
intent: If media origin remained unresolved, re-examine the media for additional
contextual clues (visible dates, locations, signage) to reconcile it with
the claimed context.
query_guidance: Compare specific keyframes side-by-side with verified footage
of the candidate event to confirm they depict the same moment (EXACT) rather
than merely similar (PARTIAL). If still unresolved, report as unverified.
activates_checks:
- corroboration_obtained
- counterevidence_checked
id: layer2_corroborate
transition:
- if: continue
to: layer2_synthesis
type: actions
- activates_checks: []
id: layer2_synthesis
transition:
- if: continue
to: finalize
type: synthesis
start_node: layer0_origin

```

Listing A.4: offer_authenticity, the second unmerged cluster, covering 68 training claims. The smallest blueprint that carries traffic.

```

description: Verifies online commercial or transactional solicitations - offers, promotions,
giveaways, recruitments, subsidies, surveys, prices, and brand/agency attributions
of such solicitations - by resolving destination domains, checking the named entity's
official channels for the offer and for any warning/denial, and cross-referencing
templated scam patterns. Confirming a legitimate offer is as valid an outcome as
identifying a fraudulent one. Distinct from all other blueprints in that the decisive

```

```

step is domain/URL resolution and matching against known scam templates, not media
forensics, quote/record tracing, or event corroboration.
entry_conditions:
  all:
    - feature: has_claim_text
      op: ==
      value: true
  any:
    - feature: asserts_recontextualization
      op: ==
      value: true
    - feature: asserts_identity
      op: ==
      value: true
    - feature: is_quote_attribution
      op: ==
      value: true
    - feature: asserts_place_or_date
      op: ==
      value: true
name: offer_authenticity
policy_constraints:
  allowed_actions:
    - web_search
    - media
  max_iterations: 5
  require_counterevidence_search: true
required_checks:
- description: The destination URL/domain of the offer was resolved and its registration
  date, ownership, and relation to the official domain were checked (a domain matching
  the official entity supports authenticity; a recently-registered look-alike or
  unrelated domain supports fraud).
  id: destination_domain_resolved
- description: The official website and social media channels of the named brand,
  agency, or government body were checked for the offer AND for any warning or denial.
  Absence of the offer on official channels, or an official denial, was recorded;
  presence of a genuine official announcement is an equally valid finding.
  id: official_channels_checked
- description: A direct statement from the named entity (press office, customer service,
  or official post) was sought regarding whether the offer is genuine.
  id: entity_statement_sought
- description: The account or page hosting the offer was analyzed for creation date,
  transparency/admin location, name-change history, and follower count where applicable.
  id: hosting_account_analyzed
- description: The claim was cross-referenced against prior fact-checks, security
  warnings, or identical templated offers circulating in other regions/countries.
  id: cross_reference_checked
selector_hints:
  negative:
    examples:
      - This video was generated by AI - goes to media_document_attribution_provenance_and_synthetic_origin.
      - A study found that a drug reduces mortality by 30% - goes to
primary_source_official_scientific_statistical_and_attribution_record_verification.
      - A statistic or official policy figure with no promotional offer or solicitation
      - goes to primary_source_official_scientific_statistical_and_attribution_record_verification.
    features:
      - asserts_synthetic_origin
      - is_scientific_medical
      - is_statistical
  positive:
    examples:
      - A viral post claims a well-known brand is giving away free products if you fill
      out a survey and share the link on WhatsApp.
      - A website advertises a government cash subsidy for a specific group, asking
      for personal and banking details.
      - A Facebook page announces a recruitment drive for a public agency with mandatory
      registration fees.

```

```

- An official body reportedly warned that a circulating promotion or giveaway
  link is a phishing scam.
features:
- has_claim_text
- has_url
- has_image
- is_multimodal
verification_graph:
nodes:
- actions:
- action: web_search
  intent: Identify the named entity (brand, agency, government body, or person)
    and locate its genuine official website and verified social media channels.
    Determine whether the advertised offer, price, promotion, recruitment, or
    statement appears on those official channels - its presence supports authenticity,
    its absence is a meaningful but non-conclusive finding.
  query_guidance: Search for the entity's official name plus 'official site',
    'careers', 'promotion', 'aviso'/'warning'/'scam'/'estafa'/'fake' in the relevant
    language. Note the official domain (e.g. .gov.xx, brand's own domain) for
    later comparison.
- action: web_search
  intent: Resolve the destination URL/domain the offer points to and check its
    registration date and ownership; compare it against the entity's official
    domain. A domain that matches the official entity supports authenticity; a
    recently-registered look-alike, generic, or unrelated domain supports fraud.
  query_guidance: Perform or search for WHOIS/registration data on the destination
    domain. Compare the domain string and registrar against the official domain
    identified above. Note registration date, registrar, and country.
- action: media
  intent: If an image is attached (e.g. a product photo, prize photo, or branded
    graphic), establish its true origin and compare it with the claimed context.
    A reverse-image or stock-site match counts against the media only if it is
    an EXACT match of THIS image; PARTIAL or look-alike matches, and matches to
    other items, are not evidence about this media.
  query_guidance: Reverse-image search the attached media; check for exact matches
    on stock sites (Pixabay, etc.), marketplaces (eBay, Light in the Box), or
    the official brand catalogue. Use the EXACT vs PARTIAL label on each match
    - only EXACT matches speak to this image's origin.
activates_checks:
- destination_domain_resolved
- official_channels_checked
id: layer0_scope
transition:
- if: continue
  to: layer1_synthesis
type: actions
- activates_checks: []
id: layer1_synthesis
transition:
- if: the offer is clearly confirmed as genuine on official channels, or clearly
  refuted by an official denial and a look-alike/unrelated destination domain,
  with consistent evidence
  to: finalize
- if: the entity's own position is not yet established, or the hosting account
  and cross-region context remain uninvestigated, or evidence is mixed
  to: layer2_deepen
type: synthesis
- actions:
- action: web_search
  intent: Seek a direct statement from the named entity about whether the offer
    is genuine - a press-office comment, customer-service confirmation, or an
    official post. Record either a confirmation that the offer is real or a denial/scam
    warning; both are valid outcomes. If no statement can be obtained, report
    that aspect as unverified rather than assuming fraud.
  query_guidance: Search for official press releases, media statements, or verified
    social posts from the entity referencing the offer. Include national cybersecurity/consumer-protection
    agency warnings where relevant.

```

```

- action: web_search
  intent: 'Analyze the account or page hosting the offer: creation date, page-transparency/admin
    location, prior names, follower count, and posting history. Also cross-reference
    the offer against prior fact-checks and identical templated offers in other
    countries or regions.'
  query_guidance: Look up Facebook Page Transparency / creation dates, search
    for the offer's exact wording plus other country names or currencies, and
    search fact-check archives and security reports (e.g. Cybertrace) for the
    same template.
  activates_checks:
  - entity_statement_sought
  - hosting_account_analyzed
  - cross_reference_checked
  id: layer2_deepen
  transition:
  - if: continue
    to: layer2_synthesis
  type: actions
- activates_checks: []
  id: layer2_synthesis
  transition:
  - if: continue
    to: finalize
  type: synthesis
start_node: layer0_scope

```

Listing A.5: generic, the fallback blueprint. It is not synthesized from a cluster and absorbs claims the selector cannot route, which is why its entry conditions are empty and its iteration budget is the smallest in the pool.

```

description: General-purpose fact-checking strategy for arbitrary multimodal claims.
entry_conditions:
  all: []
  any: []
name: generic
policy_constraints:
  allowed_actions:
  - web_search
  - media
  max_iterations: 4
  require_counterevidence_search: true
required_checks:
- description: The main factual content of the claim was investigated.
  id: core_claim_checked
- description: Counterevidence or contradiction search was attempted when relevant.
  id: counterevidence_checked
selector_hints:
  negative:
    examples: []
    features: []
  positive:
    examples:
    - A politician said X happened in 2024.
    - This image shows a protest in Athens.
    - This video was filmed during the election campaign.
    features: []
verification_graph:
  nodes:
  - actions:
    - action: web_search
      intent: establish the core factual context and locate likely primary sources
      query_guidance: null
    - action: media
      intent: investigate any attached media for authenticity, origin, or context
      query_guidance: null
  activates_checks: []

```

```
id: layer0_scope
transition:
- if: continue
  to: layer1_expand
type: actions
- activates_checks: []
id: layer1_expand
transition:
- if: core claim is already verified or refuted with sufficient evidence
  to: finalize
- if: evidence is insufficient, contradictory, or key aspects of the claim remain
  uninvestigated
  to: layer2_deepen
type: synthesis
- actions:
- action: web_search
  intent: follow up on open questions identified in the first synthesis
  query_guidance: address the specific aspects of the claim that were unresolved
  or weakly sourced
- action: web_search
  intent: search explicitly for counter-evidence and contradictions to the claim
  query_guidance: look for corrections, fact-checks, or sources that contradict
  the evidence found so far
activates_checks: []
id: layer2_deepen
transition:
- if: continue
  to: layer2_synthesis
type: actions
- activates_checks: []
id: layer2_synthesis
transition:
- if: continue
  to: finalize
type: synthesis
start_node: layer0_scope
```

A.8. The Learned Strategy Document

Listing A.6 reproduces Strategy.md in full, exactly as the single-strategy planner of Section 5.1.4 receives it. Markdown markup appears verbatim, since the document is threaded into the system prompt as raw text and never rendered. Section 4.5.1 describes the construction procedure that produced it. The five occurrences of != in the general principles read ≠ in the source file.

Listing A.6: Strategy.md, the distilled playbook used by the single-strategy condition of Section 5.1.4. Built by CLAUDE OPUS 4.8 over the 3,709 training article analyses in one epoch of 371 folds, and roughly 4,300 words in eight sections.

```
# Strategy.md - Fact-Checking Playbook

> A living playbook of reusable *methods* for verifying claims - guidance, not script.

## General principles

- **Every check runs both ways.** The tool that debunks a fake confirms the real thing. Confirming truth is a valid outcome - an accurate quote/statute/statistic, a real event, a routine procedure, a settled null, or something *discussed but never implemented*. Don't over-index on "false."
- **The responsible body's own record settles claims about its acts.** A body flagging something fake, lacking any record of a claimed act, or confirming it knows of *no* credible evidence is confirmable evidence.
- **A preliminary finding can be superseded.** An alert/field-test/estimate later corrected by confirmatory testing or an official update - seek the *latest* status from the issuing body.
- **Attribute the act to the body that holds the power, at the right stage:**
  - **Stage gaps** - proposal != adoption; bill != law; filing != decision; call != done; plan != present.
  - **Competence gaps** - a supranational body can't decide what's reserved to a member state.
  - **Tenure gaps** - a decision or trend predating a person's entry to office isn't theirs.
- **Self-declared metadata is not authoritative** - dates/timestamps/bylines may be user-set or masked; a null or fictitious byline is a fabrication tell.
- **Absence of evidence is evidence when coverage would be expected.** A thorough null across mainstream media, official channels, registries, scientific literature, *and* all implicated parties' channels disconfirms - and confirms a "no such thing exists / no evidence for X" claim, especially when an expert body states it knows of none.
- **Separate the true kernel from false amplification.** A partly-true base doesn't validate inflated numbers, invented motive, broadened scope, a misattributed actor/period/location, a conditional rate as baseline, single-factor causal spin, or a false "now/here/first." Render component verdicts, including "imprecise/misleading."
- **An allegation is not a fact.** Claims resting on anonymous sources, a single accuser, undisclosed-methodology surveys, admitted extrapolation, speculative forecasts, propaganda outlets, a flawed small-sample study, or a discredited originator stay *unproven* - never confirmed by repetition. Trace to origin; check for recantation or larger replication.

## Approach by claim type

*Each subsection names the decisive evidence; see **Evidence techniques** for mechanics.*

### Media authenticity (video/audio/image manipulation & AI)
- **Establish the original, per element.** Trace each image, clip, background, and audio track separately; split compilations and render element-level verdicts.
- **A creator/platform self-label or admission settles it both ways** - watermark, AI-tag/SynthID, generator/effects logo, creator-tool tags, platform disclaimer, satire/prank byline, "About" page, hashtags, or a caption confirming AI /CGI/staged/satire/the real subject. A meme-maker's follow-up admitting an edit is decisive; check the uploader's *other* posts for a CGI/skit/dramatized history.
- **A viral image often alters a *real* record** (face swapped, object added, subjects spliced), or a deepfake fabricates a compromising *action* over a benign event - pull the authentic original and compare pose-for-pose.
- **Decisive evidence by sub-type:**
  - **General fake:** located unmanipulated original, or provider-confirmed generator watermark (strongest - beats subtle tells and split detector scores).
  - **Fabricated post/quote-card/form/ballot:** impersonated party's denial plus layout comparison against the authentic template.
  - **"Segment removed":** the full archived broadcast at the cited timestamp.
  - **Staged-as-real:** a longer cut or alternative angles revealing crew/recurring actors/gear; for staged official material, the body's fake-flag plus comparison against genuine records.
  - **Audio superimposition:** the audio circulating before the footage.
```

- ****Accurate clip (confirm):**** earliest crediting broadcast with matching location features plus translated in-audio narration matching an official source.

Context manipulation (real content, false framing)

The most frequent failure mode; treat framing as the claim. Date, geolocate, and read in-frame/in-audio evidence first - framing may be ***correct***. ****Fast tells:**** caption contradicting a visible label/landmark/watermark; wrong spoken language for the claimed country; lighting/weather mismatch; reused trending audio as live sound; a masked or cropped date.

****Framing swaps to watch:****

- ****Misattributed cause/actor/motive/period/location**** - an ordinary crime or natural event relabeled deliberate/political/communal; foreign relabeled domestic (or vice versa); a festival gathering relabeled a protest. If location is disproven but motive stays unverified, render location false and motive unproven.
- ****Crop / freeze / inversion / splice / audio-swap**** - a fuller angle or full transcript restores (and equally confirms) the truth; identify ***who each line addressed***. Sarcasm/meme intent is confirmable only from the untrimmed cut.
- ****Old recast as new**** - content online before a referenced event can't depict it; an entity can't act before its founding. A masked date often hides a genuine old item.
- ****Wrong-person presence/absence/arrest**** - disprove via geolocation ***and*** real-time status; settle via the full official broadcast/log at multiple timestamps.
- ****Scope/superlative distortion**** - a localized/conditional case recast as sweeping policy; a false "first/only"; a bill recast as law.
- ****Real-but-not-spontaneous**** - an authentic event framed as organic when organized/announced/registered; confirm the trigger via organizers and official channels.
- ****Anomaly framed as scandal**** - a routine/procedural situation spun as sinister; verify what is ***normal*** via comparable official records or an expert.
- ****Single-factor causal overreach & snapshot-as-trend**** - a policy credited with an outcome that still occurs; a decline claimed as a full stop, refuted by the issuing body's latest figures.
- ****Study/report/exercise/treaty/old-broadcast reframed as official plan or current event**** - trace the underlying document/clip; read its scope/date/audience; get the body's on-record status.

Identity & actor-attribute claims

- ****Fix true identity, role, and affiliation before accepting any label.**** Mismatch in rank/office, party, role-at-the-time, community, nationality, insignia, relationship, faith, or living/deceased status flags misidentification. The ****authoritative list/roster settles a candidacy/office/membership/registration claim both ways****; a viral portrait is often an unrelated individual, lookalike, or actor. ****The subject's own repeated, consistent self-identification or denial settles an affiliation rumor**** - cross-check against the original allegation.
- ****Communal/religious/xenophobic spin**** - verify each party's actual faith/community/origin and the real dispute before accepting a communal framing.
- ****Endorsement/testimonial**** - reverse-face-search the pictured "expert" (often stolen); check the named professional against the licensing registry (null = not real); fabricated endorsements are often disavowed on the named person's channel.
- ****Impostor page/account**** - page-transparency history (prior names, recent creation, mismatched contact) exposes it; the real party's verified account and warning post are decisive both ways.
- ****Weigh the originator's track record.**** A promoter with a suspended license, agenda, disinformation history, or parody label raises the bar or settles the origin.

Event, death & crime claims

- ****Check for expected mainstream coverage**** plus official/family/institution channels. Multiple independent credible reports (incl. local-language outlets naming parties and citing an official statement) confirm; a direct denial settles a rumor. A claimed death/crash/kidnapping is often refuted by the operator's statement or continued public activity; where even close associates can't confirm, it stays ***unproven***.
- ****A claimed unit/group/product/ministry/event/business with no footprint doesn't exist.**** A convergent null across cited outlet, alleged actor's channels, official portals, and marketplaces refutes; a defunct or not-yet-operational entity can't issue current material. Past ***discussion*** of a never-launched venture doesn't make current existence true. Conversely, an organization's own materials and spokesperson confirm it genuinely holds a stated benign position.
- ****Ban/decreed/coup/emergency/arrest/inquiry/new-ministry/deployment rumors**** - the named authority's on-record statement or published document settles it both ways (verify document authenticity). For service bans, add users confirming it works plus a news null. Often traceable to a prior unimplemented ***call*** or recycled older audio.
- ****Blame/negligence/refusal allegations**** - trace to origin (often partisan); the accused's spokesperson plus the coordinating authority's log/alerts can show the act was authorized, done as directed, or never occurred. Note which body holds jurisdiction.
- ****State-media / propaganda allegations**** - an accusation aired only by a partisan/anonymous sourceless channel stays ***unproven***; a claim traced to a foreign disinformation network is a strong fabrication signal; the accused's denial plus a role-timeline refutes.
- ****Foreign-funding / procurement / weapons / ownership claims**** - the official registry/filing naming only other

sources or confirming unchanged ownership refutes; the company's own filings/denial settle a claimed change both ways.

- ****Historical, biographical & etymological claims**** - settle via period archives, official documents, dated first-use research, domain historians; a modern origin story is implausible when the practice predates the term's first attestation.
- ****Scientific/natural-event & resource claims**** - verify severity/nature/toll against the responsible monitoring agency's scales/database/report (plus an expert), independently of any attached footage. An "unnatural" phenomenon is refuted when the effect is a documented natural process. For extreme-weather/climate causal claims, use rapid-attribution and IPCC-type assessments plus specialists; separate the *weather driver* from *infrastructure/logistics* sub-claims. An operator ruling out a cause, or confirmatory testing correcting a field result, settles a causal rumor even while a broader inquiry continues.
- ****Feasibility & chronology test**** refutes the physically, procedurally, legally, arithmetically, chronologically, or biologically impossible - including a security mechanism making the act impossible.

Existence, ownership, charity & suspect offers

- ****Disprove existence at the authoritative index**** - registry, marketplace, license/drug-approval database, portal, tracker, gazette, spending record, docket, roster, bibliography. A live listing confirms; a null plus denial refutes. Lookups may require the subject's legal name. Check whether a claimed official act is actually a *self-service* product anyone can create.
- ****Fundraising / charity / recruitment / scholarship legitimacy**** - convergent multi-signal test:
 - verify tax/registration IDs (invalid ID or many shells at one address = fraud tell)
 - geolocate the registered address (an ordinary residence undercuts a large NGO)
 - check domain/page age
 - search for identical template campaigns swapping only names/countries
 - confirm the offer/form against the backing institution's official channels (mismatch or public "fake" flag refutes)
- ****Program/aid/tariff/budget/tax-scope & procedure claims**** - confirm exact eligibility, amounts, scope, steps from the issuing body's documentation/form/spokesperson/draft law. Watch scope-inflation, figures exceeding a hard ceiling, and conditional rates applying only above a quota/trigger.
- ****"Hidden device/feature does X with your data"***** - identify the object against teardowns (usually mundane); settle data/privacy claims by the provider's official FAQ/terms.
- ****Suspect offer / phishing / endorsed scam ad**** - any single failure marks it untrustworthy: no such offer on the backing institution's channels; endorser's disavowal; unrealistic terms or upfront payment; empty/off-topic linked page; off-domain/lookalike/flagged URL; foreign-based admins; no license.

Quote & source attribution

- ****Get the exact original at the timestamp.**** The complete original on the speaker's own channel plus event video/transcript at the cited mark is decisive - and equally *confirms* an accurate quote or the *absence* of a claimed statement. It exposes:
 - quoting-to-condemn, sarcasm, paraphrase, spliced quote, inserted line, stripped hypothetical, a line re-pointed at the wrong interlocutor
 - fabricated subtitle, mistranslation, inverted meaning, cloned/dubbed voice
 - a ****softened phrase hardened**** (measured concern recast as a threat)
- ****Preserve the attribution.**** An allegation reported "say opponents" is falsely relayed when the qualifier is dropped and stated as the outlet's own fact - retrieve the original headline verbatim. A clickbait headline often contradicts its own article body.
- ****Verify a cited document/statement actually applies.**** A claim may rest on a real statement about a ***different*** subject/scope; an old article may be relinked under a false new headline.
- ****A viral claim often distorts a real interview/post; the original poster frequently denies the viral framing.**** Check edit-history; watch for later deletion.
- ****Satire/fiction/prank/skit is a recurring true origin**** - a disclaimer/"About" page, parody bio, recurring-actor channel, or fictional "spokesperson" settles a "staged/satire" claim; cross-check the person's actual statement.

Statistics, poll, scientific, legal & institutional claims

- ****A poll/survey claim is unfounded without disclosed provenance**** - no pollster, methodology, sample size, or fieldwork date = unsupported; cross-check against credible published polls and historical data.
- ****Go to primary documentation and domain experts; recompute.**** A figure in no official document is fabricated; a body confirming figure *and context* confirms it. A multi-part claim is confirmed only when ***every*** part checks.
- ****Confirm the true frame of a real figure**** - exact timeframe, age group, population, source. A striking statistic may be genuine but driven by long-term trends, not the viral spin; a trend predating an alleged cause refutes the causal frame.
- ****Rebuild a proportion from the full population**** - enumerate every member from an authoritative roster, classify, recompute; a headline count exceeding the true population flags exaggeration.
- ****A quoted figure may be real but mislabeled**** - read the exact question; a max/total posed as an average distorts. A ranking is confirmable only when its method is disclosed (raw vs per-capita vs adjusted).
- ****A "most ever / record" superlative demands recomputation across all comparable periods**** using the issuing body's database; beware a proponent's non-standard counting excluding comparable items.
- ****Test a document's contents by searching its full text.**** A keyword null inside the real document, plus the correct

provision, refutes a misdescription; absence of a prohibiting/authorizing provision confirms a "not there" claim.

- ****Trace a striking figure to its actual derivation.**** A viral number often comes from a third party's arithmetic on raw self-report data that **cannot establish causality** (e.g. adverse-event report systems), possibly resting on a later-corrected error. Author disavowal confirms "misrepresents the study"; endorsement confirms accuracy.
- ****Weigh study quality, not just existence:****
 - read its actual cause/effect and timeframe; data predating the alleged cause refutes causation
 - a ****large cohort/sibling-comparison finding no association is strong confirming evidence for a null****
 - a small-sample or preclinical/early-stage study is not evidence of a human effect - check for larger replication and the author's stated limits
 - a trial stopped early may reflect a **better alternative**, not harm
- ****A settled consensus is confirmable, not automatically "overstated."**** Converging methodologies, meta-analyses, and authoritative-body reports plus a topic expert confirm it; local/short-term outliers don't refute a long-term global trend.
- ****Data-granularity limits can leave a claim unprovable**** - when the issuing body confirms it doesn't collect data at the claimed breakdown and the proponent won't supply their analysis, the claim is **unproven**.
- ****Denominator & comparability are core tests:**** an adverse-outcome count means nothing without the group total; a raw headcount as "most" is refuted by the true total; report-counts prove neither incidence nor causality; figures aren't comparable across differing scope, method, unit, vintage, population, or stage.
- ****Causal inversion**** - risk of a **condition** is routinely flipped into risk of its **treatment/prevention**; check what the study measured.
- ****Read the actual legal/policy/treaty/statutory/procedural text**** for scope, basis, binding force, and **competent branch/level**. Distinguish administrative vs substantive, proposed vs approved, voluntary vs mandatory, advisory vs binding. A statute's text **lacking** a claimed authorization confirms a "the law doesn't permit X" claim.

Health, nutrition & technical-mechanism claims

- ****Anchor to peer-reviewed literature, multiple named clinical experts, and official regulatory/dietary guidance**** - decisive both ways; don't default to skepticism against established guidance or physiology. Corroborate with the national drug/medical authority's current recommendation.
- ****"Limited/preliminary evidence" is often the accurate verdict.**** A hedged claim (some effect but **lacks strong clinical support**) is confirmable as-stated. Match the evidence bar to scope and stage: a human-effect claim needs human evidence; a preclinical drug is not an approved treatment.
- ****A nuanced "does X but not Y" medical claim is often precisely true**** - confirm each half against clinical sources rather than treating the whole as over-claim.
- ****"Cures/reverses a chronic, autoimmune, cancerous, or irreversible condition" is a red flag.**** Where authorities define a condition as incurable or no clinical evidence supports the cure, no diet/remedy/repurposed drug overrides it - but **standard management** and **modest adjunct effects** are independently confirmable.
- ****"A common ingredient/gene variant is a toxin" alarm**** - check official guidance on whether the population processes it and whether the named factor is genuinely listed; match evidence to dose and whole-food context.
- ****Product-spec / mechanism claims**** - confirm against the maker's official specs and patents plus a domain expert; a viral "how it works" often garbles a real, mundane mechanism.
- ****Alarming measurement / emission claims**** - verify the **instrument** (uncalibrated = unreliable), then convert the reading to the official limit's unit before judging.

Evidence techniques

- ****Reverse image/video search & date-location establishment**** - the workhorse for debunking and confirming context. Split video into keyframes; locate **earliest** originals, detect recycled clips, identify who/what is pictured, prove age. Search across engines/regional platforms and native-language keywords; use face-search, mine comments, follow any in-frame identifier. Pin date/place via signage, dialect, plates, currency, livery, landmarks, geotags, Street View/satellite, weather, packaging, seasonal cues. ***Strong:*** an exact earlier source, credited creator, or matched in-frame detail. ***Weak:*** only copies of the viral version, or null.
- ****Satellite / map imagery verification**** - locate coordinates; use ****historical imagery layers**** to date the capture and test whether a feature is persistent or transient. Corroborate with a geospatial expert and local journalists. Treat maps as snapshots.
- ****Full-broadcast/keyframe & multi-angle matching**** - match to the complete original at the exact timestamp(s), using gestures/gaze/audio as anchors, to expose cropping, freezes, splicing, reversal, audio-swaps, dubbed words, fabricated subtitles, a deleted segment, or stripped context - or to confirm a faithful excerpt, a person's presence, or a film crew. Identify each interlocutor to attribute lines correctly.
- ****Foreign-language audio/text handling**** - translate in-video narration/captions (ideally via a native speaker); match word-for-word against the official release/transcript/local report to confirm origin or expose a mistranslation / fabricated subtitle. The spoken language can itself disprove a claimed location.
- ****Doctored-screenshot / forged-document comparison**** - find the original article/edition/post/listing and compare format-for-format; fakes are often cloned from a real same-day item or hide the real date. A keyword search of the alleged publisher's site (and Wayback) refutes a "they published X" claim. Authenticate an official document by matching signatory, dateline, format, channel. ***Forged tells:*** wrong/missing/misspelled signatory, date, or dateline; unofficial channel; off-domain URL; wrong fonts/layout; typos; satirical byline.
- ****Object/uniform/prop identification**** - match design, scale, insignia, flags, attire, vehicle model against

manufacturer/ministry specs or a verified reference.

- ****Document/regulation/full-text lookup**** - retrieve the official document, draft text, form, docket, tender, treaty-party list, report, transcript, or law/statute/order by title/article/number; confirm true audience, mechanism, scope, competence, currency. Read what it **does** say vs the attributed claim.
- ****Archive / Wayback & edit-history lookup**** - snapshots prove **when** a page/post/dataset appeared, changed, or vanished; reveal a "new" status unchanged for years; recover a deleted post; expose the earlier version where a distortion originated; confirm the **absence** of a claimed statement. Forwarding trackers can date a viral item to a specific (often partisan) channel.
- ****Publisher/platform metadata & transparency verification**** - confirm true publication time and identity when a displayed date/author may be user-set. Page/domain creation date, prior page names, follower count, admin location, ownership expose a fresh imposter page, content farm, or foreign-run scam/disinformation network.
- ****Image forensics & AI/deepfake/CGI detection**** - treat detector scores as corroborating; cross-check multiple tools (video and audio separately) plus platform AI/watermark/SynthID/generator-logo indicators and expert analysis. A tool logo, provider-confirmed watermark, admission, creator-tool tags, or located original beats a split score. Same-prompt reproduction is strong; ELA localizes edits. **Tells:** extra/missing fingers or deformed anatomy; incomplete/morphing figures or text; distorted background; splicing lines; blurred edges/halos; waxy skin; impossible shadows/light/reflections; static figures; cloned/dubbed voice; missing expected logo/microphone; lip-sync mismatch; production equipment.
- ****Reverse face / voice comparison**** - check whether a "person" is a stock model, different individual, stolen/AI portrait, lookalike, actor, or promoter posing as expert; compare voice against dated authentic recordings.
- ****Stock/press/licensing-library asset lookup**** - an agency watermark, asset ID, or caption leads to the unedited original and its true date/location/subject.
- ****Primary-record & provenance lookup**** - registries, business/tax-ID records, finance/donor/procurement registries, oversight reports, dockets, statutes, vote/pardon records, licensing directories, filings, stock announcements, casualty rosters, census/labor data, office-holder rosters, obituary/family/court records, denomination/religion registers, national archives. Cross-check serials; search under legal names.
- ****Domain/URL & account-origin forensics**** - WHOIS/creation date and a malware scanner flag a fresh, lookalike, or malicious site. Sandbox suspect links; inspect page-transparency/admin data. **Impostor/scam tells:** thin/renamed/same-day page; misspelled name or off-by-one handle/domain; non-official TLD; free-mail vs official domain; copied logo or AI profile picture; mismatched contacts; foreign-based admins.
- ****Primary-dataset recomputation**** - reproduce the cited figure by recipient/category/timeframe/population/unit; obtain the full denominator; enumerate a defined group and reclassify; test against a hard ceiling; adjust for inflation/confounders; fix baseline-vs-endpoint; recompute a "record" across all comparable periods. Read the exact survey question. Trace a derived figure through intermediary arithmetic, noting when it rests on data that cannot establish causality.
- ****Direct authority/subject/expert/journalist contact**** - an on-record statement, retraction, denial, admission, or "we know of no evidence" from the named operator, regulator, spokesperson, court, company/author, family, physician, lawyer, local journalist, organizer, ordinary local user, uploader/photographer, or domain expert is among the strongest evidence either way. Use **multiple independent** experts for technical/health/legal claims. A non-response leaves that point unresolved but doesn't override other decisive evidence.
- ****Live operational & hazard signals**** - flight/transit/vessel trackers, official incident/fire/weather/AQI/evacuation maps/alerts, webcams, schedules, and body calendars determine whether a service/event is running, blocked, routine, or at claimed scale - and can place a person or landmark elsewhere in real time.
- ****Verbatim quote/headline & site-scoped multilingual search**** - surface true origin, wording, and tone across the subject's language, the audience's, and English; run a site-specific search of the cited domain plus verified accounts and the primary text. A total null disconfirms a fabricated headline/order/statistic/quote/poll; a match confirms it. A year-restricted search refutes a "recent event" with no coverage. Re-verify a prior fact-check's primary evidence rather than echoing its verdict.

Common pitfalls

- ****Overshooting to "false"***** - dismissing genuine content: a real quote/event/statistic/ruling/study/spec/statute; an accurately-captioned or true old-footage clip; a standard/in-force or **routine** procedure; a settled consensus, established medical management, or a large-study null; a correctly-hedged "limited evidence" or nuanced "does X but not Y" claim; a documented natural process; an organization genuinely holding a benign position; a registered/organized event; a thing genuinely discussed but not yet implemented.
- ****Over-trusting tools/summaries**** - a single/split detector score, plausibility, "peer-reviewed" label, self-declared metadata, uncalibrated instrument, small-sample/preclinical study, or AI/search summary treated as proof without a located original or second signal.
- ****Echoing without re-checking**** - a reputable outlet's error; an op-ed as official data; a deleted/retracted/speculative source; propaganda, anonymous-channel, or disinformation-network claim as reporting; a dated screenshot/clip as current; recycled audio/footage as live; a parody/prank/skit/AI/CGI page taken literally; a source describing a different subject or the opposite; a copy-paste rumor swapping only the named subject; a clickbait headline contradicting its body; an unimplemented **call** reported as done; an allegation relayed without its "say opponents" attribution; an initial field result after correction; a quote-of-a-false-post read as the speaker's own view; a derivation from causality-incapable self-report data.
- ****Treating a preliminary step as settled**** - an allegation/hypothetical/extrapolation/draft/pilot-study/non-binding

strategy/anonymous forecast as fact, law, or plan; a discussion as a decision; a decline as a full stop; a plan/filing as accomplished; an advisory opinion as binding.

- ****Statistical/study traps**** - mismatched scope/method/unit/vintage/population/stage; a snapshot as a trend; a local outlier as a global counter-trend; dropped denominators; a raw headcount as "most"; cases as people; report-counts as incidence; a confounded comparison read as causal; a conditional rate as baseline; data predating the alleged cause; one factor credited for a whole outcome; a condition's risk flipped into its treatment's; a figure exceeding a ceiling; a "record" via non-standard counting.
- ****Attribution/scene misreads**** - bystander vs described actor; a portrait/lookalike/namesake/actor vs the real figure; impostor page vs real official; staged/filmed scene vs real event; a real photo with a swapped face or fabricated action; an organized one-off vs an organic symptom; living vs deceased; role-now vs role-at-the-time; a false communal/xenophobic/political/religious spin on ordinary activity; footage of one place/event relabeled as another; a natural event relabeled deliberate; an act attributed before the actor existed/took office; AI-audio dubbed over authentic footage; a line re-pointed at the wrong interlocutor; a national-competence matter attributed to a supranational body; an authorized/directed act spun as negligence.

When to stop

- ****Convergence**** - independent lines (primary source, official confirmation, multiple experts/journalists, detection tools, recomputed data, live/satellite signals) agree.
- ****Decisive primary evidence, in any direction:****
 - a located earliest original contradicting *or confirming* the framing, date, place, subject, language, or altered element - incl. a source predating the referenced event, a matched detail, dated satellite imagery, a matching broadcast, or the genuine record a viral image altered
 - an on-record authority/subject/author/family/expert/journalist/organizer/uploader/local-user statement or official record (own channel, authoritative roster, a body flagging something fake, lacking any record, or confirming it knows of no evidence; an operator ruling out a cause; confirmatory testing correcting an initial result)
 - exact statutory/ruling/vote/budget/tender/treaty/study/spec/transcript text (or a full-text null); a reproduced statistic with its true actor/period/baseline/denominator/unit/question; or a roster-enumerated proportion
 - a registry/marketplace/archive/roster/docket/approval/tax-ID/procurement/census/WHOIS/denomination-register entry pinning origin/date/ownership/identity/existence (or a null where it would exist)
 - a physical/legal/arithmetic/chronological/biological/security-mechanism impossibility; a dataset predating the alleged cause; confirmed consensus/mechanism/standard medical management/natural process/large-study null; or a competence/jurisdiction mismatch
 - an admission or clear origin of fabrication/satire/prank/AI/CGI/VFX/staging - a generator/effects logo, creator-tool tags, provider-confirmed watermark, voice-clone/dub, reversal, ELA finding, same-prompt reproduction, creator's own "I made this meme" post, satirical/skit disclaimer, recurring-actor/dramatized channel, or a lookalike/content-farm/scam-domain/disinformation-network/recycled-hoax/renamed-page origin
- ****Expected-but-absent coverage**** - for high-visibility claims, a diligent multilingual negative search across mainstream, fact-check, official, court, marketplace, scientific-literature, and *all implicated parties'* channels suffices to reject (or confirm a non-existence / no-evidence claim).
- ****Component-level resolution**** - for compound claims render a mixed or "imprecise/misleading" verdict (real footage / false framing; real event / wrong date-place-actor-scope; real but organized-not-organic; false headline / genuine stance; real quote / inverted-hardened-or-misaddressed meaning; real number / non-comparable metric; real statute / inflated authorization; real decline / overstated as full stop; genuine effect / no cure).
- ****Declare unproven - not false -**** when a widely-cited figure or poll has no disclosed methodology or official tracking, evidence rests on a single dubious/propaganda/speculative/small-sample source, detection is ambiguous without a located original, the attribution chain is unresolved, an identity awaits forensics, or a key authority hasn't responded (while no other decisive evidence resolves it).