Designing an Automated Fact-Checking Pipeline for Recent Claims

Arshia Arya
University of California, San Diego
San Diego, California, USA
aarshia@ucsd.edu

Gauthami Yenne*
Georgia Institute of Technology
Atlanta, Georgia, USA
gyenne3@gatech.edu

Deepak Kumar
University of California, San Diego
San Diego, California, USA
kumarde@ucsd.edu

Abstract

To address the scale and spread of digital misinformation, researchers have developed automated techniques to aid in the fact-checking process. Unfortunately, these systems are rarely designed for real-world deployment settings and struggle with the unique challenges of realistic verification: claims may lack ground truth, evidence is sparse and evolving, and decisions must be made within hours. In this paper, we present and evaluate a new system for automated fact-checking, RealTiCheck, that integrates large language models (LLMs) with web search to gather, collate, and employ online evidence to fact-check recent potential misinformation claims. We evaluate RealTiCheck on state-of-the-art benchmark datasets, with our system outperforming comparable prior work (Macro-F1 score of 0.81). We then explore how RealTiCheck can be used in practice by hiring seven professional fact-checkers to fact-check 174 novel and recent (less than 24 hours old) claims. We evaluate how well our system triages and adjudicates claims compared to professional fact-checkers' verdicts. We conclude by discussing design recommendations for future real-time fact-checking systems.

CCS Concepts

• **Security and privacy → Human and societal aspects of security and privacy**; **Social aspects of security and privacy**.

Keywords

Automated Fact-Checking, Web Search, Large Language Models, Claim Verification

ACM Reference Format:
Arshia Arya, Gauthami Yenne, and Deepak Kumar. 2026. Designing an Automated Fact-Checking Pipeline for Recent Claims. In *Proceedings of the 2026 ACM SIGSAC Conference on Computer and Communications Security (CCS '26), November 15–19, 2026, The Hague, Netherlands.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3830454.3832654

1 Introduction

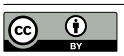
Fact-checking is an important defense in the fight against misinformation, propaganda, and outright falsehoods on the Internet. Indeed, a vast infrastructure of journalists [24], newsrooms [7], third-party agencies [1, 2], and even end-users [58] contribute to fact-checking to contextualize and correct falsehoods before they spread widely [37]. Unfortunately, traditional fact-checking requires significant human labor and time, taking hours or even days [3], and thus can rarely keep up with the scope and scale of digital content today [23, 82]. Even the most diligent fact-checkers rely on ad hoc prioritization of claims, which misses potentially harmful claims before they spread widely [66].

To tackle these scale and scope issues, researchers have spent considerable effort designing and building automated fact-checking tools that seek to augment the manual fact-checking pipeline [15, 16, 26, 34, 41, 52, 91]. These tools automate various fact-checking steps, such as claim detection, evidence retrieval, and claim verification, and evaluate their work against a corpus of collected evidence. Unfortunately, they are rarely developed or tested in real-world settings and suffer from three important limitations. First, most systems rely on bespoke models that require significant technical expertise to use [86], significantly limiting their deployability [37]. Second, with few exceptions [34], most systems are not end-to-end; they focus on just one aspect of automated fact-checking without tackling the entire pipeline [88]. Finally, few tools are trained or evaluated on recent claims (i.e., claims originating within the past 24-hour news cycle), which is more akin to the claims fact-checkers must deal with daily [88].

In this paper, we address these limitations through the design and evaluation of an end-to-end automated fact-checking pipeline. Our implementation, RealTiCheck, automatically queries the web to identify relevant information from news sources related to the claim and ultimately determines whether the existing evidence supports or does not support the claim. We implement RealTiCheck using large language models (LLMs) and web search, using LLMs to perform constrained, natural language tasks while leveraging web search as an external corpus of knowledge [44]. To the best of our knowledge, *RealTiCheck is the first end-to-end fact-checking system that combines LLMs with web search and is evaluated with fact-checkers in real-world contexts.*

We extensively evaluate RealTiCheck using state-of-the-art microbenchmark datasets and find that the end-to-end pipeline achieves a macro-F1 (M-F1) of 0.81, and we also evaluate how a litany of different system configurations can affect overall performance. For example, we show how different filters on source evidence can change performance, such as relying only on *mainstream news* as an evidence corpus (5.9% improvement in macro-F1), relying only on *relevant* documents as a corpus (14.7% improvement), or implementing both filters (19.1% improvement). We also explore different aggregation strategies when considering multiple pieces of evidence to check a claim, and find, surprisingly, that majority-voting often leads to suboptimal performance compared to a thresholded aggregation scheme.

*Also with University of Illinois Urbana-Champaign.



CCS '26, The Hague, Netherlands

ACM ISBN 979-8-4007-2871-6/2026/11
https://doi.org/10.1145/3830454.3832654

Next, we conduct a six-day study in which we hire seven professional fact-checkers to fact-check 174 recent claims (i.e., claims published within 24 hours), and study how well RealTiCheck's verdicts on the claims align with fact-checker verdicts with the available evidence at the time of fact-checking. We demonstrate RealTiCheck's performance at various aggregation thresholds with this new dataset and highlight how RealTiCheck can be operationalized for different priorities like minimizing false positives during routine operations or maximizing recall during high-stakes events. RealTiCheck can be tuned for such fact-checking needs, for example, for high-precision workloads (achieving a 0.88 support-class precision at a 75% evidence threshold) or for well-balanced workloads (achieving a 0.83 support-class F1 at a 25% evidence threshold). For claims with unanimous verdict agreement between fact-checkers, RealTiCheck is able to achieve a support-class F1 of 0.87 which highlights the effectiveness of our system in a real-world deployment setting.

Finally, we draw on our observations in designing and evaluating RealTiCheck and discuss its implications for future automated fact-checking tools. Our results highlight both the promise and limitations of leveraging LLMs and web search for automated fact-checking and we discuss both tradeoffs and considerations of use for deploying a tool like RealTiCheck in practice. We hope our results will provide researchers and fact-checkers with a useful framework to build upon in future research and tooling. Our code and artifacts are available at our GitHub repository: https://github.com/Arshiaarya/RealTiCheck.

2 Background & Related Work

Fact-checking is an endeavor typically conducted in journalistic contexts to verify the accuracy and context of a claim [78]. Online fact-checking involves assessing the veracity of claims circulating on platforms such as social media or news websites. Significant prior research has examined various aspects of fact-checking, ranging from manual processes that ensure a reliable result to the development of automated fact-checking systems. In this section, we detail previous work and challenges in fact-checking and discuss how we build on prior work to design, implement, and evaluate RealTiCheck.

2.1 Current Fact-Checking Paradigms

Professional fact-checking. Typically, professional fact-checkers fact-check manually in organizations such as Full Fact,[1] FactCheck.org,[2] and PolitiFact.[3] They identify and extract claims from various sources, including public statements, media content, and social media posts [19, 50, 56, 68]. Next, they go through editorial processes and consult multiple primary sources (e.g., original government reports, official statistics, or firsthand documents) [19, 50, 56], conduct expert interviews, and thoroughly analyze available evidence [21]. This rigor is effective in reducing belief in misinformation at a global scale, with its positive effect lasting weeks after exposure [57]. The warning labels that fact-checkers produce retain their effectiveness even for readers who distrust fact-checkers [49]. Due to the manual nature of the process, each verdict can take hours or days to produce [3, 50, 82]. However, professional fact-checking faces significant scalability challenges as resources are easily overwhelmed during major events like elections and public health crises, where timely verification is often crucial [9, 26, 31, 32, 34, 79] and fact-checkers cannot keep up with demands [42]. Professional fact-checkers rely on ad hoc decision-making for claim selection in the absence of formal processes or guidelines [66], where a recent study found that Snopes and PolitiFact matched only 6.5% of claims between 2016 and 2022 [42]. This lack of coordination among fact-checking organizations can lead to a lack of coverage and underutilization of fact-checking resources.

Crowdsourced fact-checking. Crowdsourced approaches towards fact-checking have appeared across social media to keep up with the scale of online platforms by delegating fact-checking processes to end-users. The most widely adopted implementation of this is Community Notes on X, which began as BirdWatch on Twitter [58] and leverages wisdom of the crowds to scale fact-checking [5] by engaging platform users to perform fact-checking rather than professional fact-checkers. While Community Notes can provide scalability and reduce diffusion of false information [70], the approach has several limitations. Contributors exhibit partisan patterns in how they flag and evaluate notes [6, 17] and crowd verdicts struggle for consistency relative to expert fact-checkers [62]. Response time from a post's creation to a note becoming visible is often too slow to intervene during a post's most viral phase [13]. Among proposed notes, only 11.3% reach "helpful" status [60], of those, just 7% cite links from professional fact-checking websites [8].

2.2 Automated Fact-Checking

Given the limits of professional fact-checking, significant research in the last decade has designed and built *automated* fact-checking systems to help scale fact-checking. These systems are typically divided into several components that mirror professional fact-checking processes: claim detection [32, 33, 54], evidence retrieval [52, 63], and claim verification or entailment [30, 39, 52]. Each component presents unique challenges and has received significant attention; however, many of these systems are evaluated on narrow and outdated datasets and are rarely tested in realistic fact-checking conditions [88].

Researchers have developed several comprehensive datasets and systems [18, 51] to support these efforts, including FEVER [74, 75], a data set with more than 185,000 claims and their corresponding evidence, and ClaimBuster [32], which focuses on identifying check-worthy claims. Most claim verification systems rely on transformer-based and graph-based approaches [48]. Some studies have taken a large language model (LLM)-based approach to claim verification [12, 84, 85].

Current automated fact-checking systems face numerous technical and practical challenges. One is the need for constantly updated knowledge bases, as systems typically require extensive datasets to function properly [26]. However, the models used are trained on widely used datasets that are often outdated or limited in scope, leading to poor generalizability of the model in real-world deployment scenarios [88]. Real-time fact-checking is particularly challenging due to the need for rapid evidence retrieval and verification from

[1] https://fullfact.org/
[2] https://www.factcheck.org/
[3] https://www.politifact.com/

evolving information sources [52]. However, few existing systems are evaluated using real-time data or in settings that reflect actual time-sensitive fact-checking conditions [88].

2.3 LLMs and Fact-Checking

Large Language Models (LLMs) like GPT-4o, Gemini 2.0 Flash, and Claude 3.5 Sonnet have emerged as promising tools for automating various stages of the fact-checking process. Studies have demonstrated that models such as GPT-3.5 and GPT-4o exhibit reasonable accuracy in fact-checking tasks [59]. Unlike traditional automated fact-checking approaches, which often depend heavily on predefined datasets, LLMs do not need task specific retraining for each new domain or task type. LLMs have the ability to incorporate external knowledge, which has been shown to produce more accurate and contextually relevant answers in fact-checking tasks [44, 59, 81, 84].

Researchers have developed automated fact-checking systems that augment LLMs to improve their accuracy, but many existing methods lack real-time information retrieval and evaluation [81, 88]. One such system is Muse [94], which augments LLMs with credibility-aware retrieval and multimodal analysis to generate corrections to social media posts with explanation and references. However, its evaluation design relies on retrospective analysis of existing posts and expert assessments of response quality, rather than measuring system performance under real-time fact-checking constraints.

3 Design Goals

Our system, RealTiCheck, seeks to fact-check recent claims sourced from the Internet. While typical automated fact-checking systems rely on complex machine learning models and require building large, offline corpora of knowledge (e.g., a pre-collected database with evidence), our use case precludes us from leveraging many of those strategies. We synthesize our design goals and constraints below:

1. Ease of Use Current automated systems often require extensive machine learning infrastructure or specialized knowledge bases, making them impractical for many potential users [92]. This technical barrier particularly affects small newsrooms, independent journalists, and fact-checking organizations with limited resources. We seek to create a system that requires minimal setup while maintaining robust fact-checking capabilities. This approach enables deployment across diverse scenarios, from individual users seeking more context on a claim they may encounter via social media to large journalistic organizations that process multiple claims simultaneously. By reducing the technical complexity of the deployment, we seek to increase the potential usefulness of RealTiCheck in practice.

2. Collection of Real-time evidence Professional fact-checkers rely heavily on recent and credible evidence to verify claims. However, current NLP-based fact-checking systems are trained on historical datasets that do not contain contexts emerging in fast-moving news cycles [95]. For example, data sets such as MULTIFC assume the existence of relevant evidence within their training data [22], making them ineffective for recent claims for which there may be limited to no relevant evidence. RealTiCheck should dynamically collect, assess and utilize the latest credible information. This capability is crucial for addressing emerging claims, particularly during developing news events where available evidence may change rapidly. By continuously gathering new evidence rather than relying solely on pre-existing datasets, our system should be effective even as the information landscape evolves [38] and also deliberately provide evidence for each and every claim being fact-checked, as RealTiCheck does *evidence-forward fact-checking by design.* Although significant effort is put into fact-checking to identify whether a claim is true or false, this is much more challenging to automatically identify for recent claims. This is because the evidence surrounding the claim may be limited and other approaches for evidence collection, such as expert interviews, are not possible with automated approaches.

3. Speed The rapid spread of misinformation requires a quick response to fact-checking. Studies show that false information can spread 6× faster than true information on social media [80], and the initial hours after a claim appears are critical for effective intervention. Traditional fact-checking processes, which can take hours or days [3], often cannot keep up with the viral spread of false information. RealTiCheck should enable fast fact-checking while maintaining accuracy. This requires fast mechanisms to obtain evidence, efficient processing of large volumes of potential evidence, and quick assessments of whether the collected evidence supports the claim.

4. Prioritizing Claims for Fact-Checker Adjudication While current automated fact-checking systems are often framed as replacements for fact-checkers, the goal of our system is to assist human fact-checkers in their work and reduce the overall volume of claims that require human review, which addresses the scalability issues with manual fact-checking highlighted in Section 2.1. RealTiCheck also produces a verdict for every input claim, ensuring coverage of claims that may be missed by modalities such as crowdsourced fact-checking, where only a fraction of submitted notes ever reach visibility [60]. As such, RealTiCheck should *triage* claims that can be automatically supported by credible evidence and defer to human fact-checkers for the outstanding claims. We empirically investigate how to achieve this prioritization in Section 6.3.

5. Verdict Reliability and Evidence Verifiability Crowdsourced fact-checking like Community Notes face well-documented limitations around verdict reliability [6, 62] and evidence verifiability, as a substantial share of visible notes do not cite supporting evidence [8]. RealTiCheck addresses these gaps by following a fixed retrieval-and-entailment pipeline whose verdicts are determined by retrieved evidence rather than by nature of the contributor, and by attaching an explicit evidence chain (retrieved documents, per-chunk entailment outcomes, and aggregation) to every verdict it produces.

4 Design of RealTiCheck

In this section, we describe the architecture of RealTiCheck, our system for automatically collecting and evaluating evidence for claims found online. At a high level, RealTiCheck takes as input a claim (sourced from the Internet), automatically generates suitable web search queries and collates relevant information from news

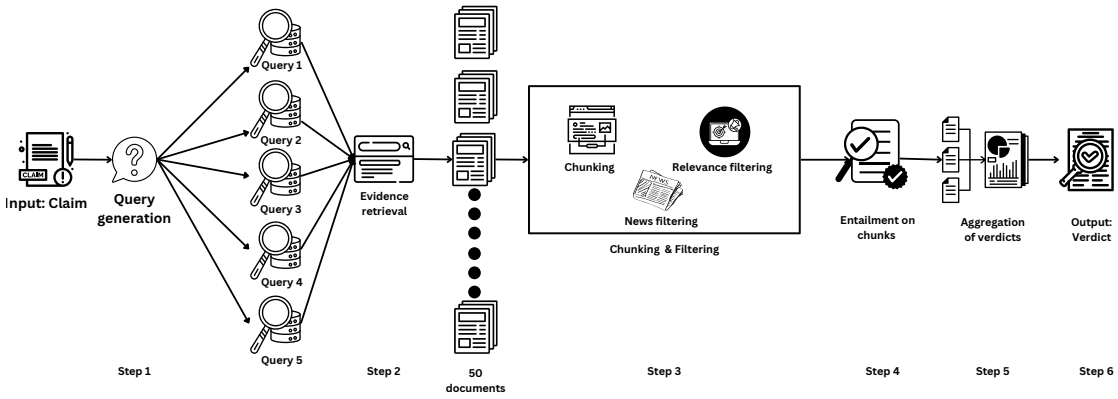


Figure 1: Fact verification pipeline: A six-step process transforming input claims into verified verdicts through query generation, evidence retrieval from multiple sources, multi-layer filtering (news, relevance), content chunking, entailment analysis, and final verdict aggregation.

sources. Finally, it adjudicates whether the claim is supported (or not) by the available evidence. Figure 1 shows the complete flow of RealTiCheck, and we evaluate each component in Section 5.

Step 1: Query Generation. Given an input claim, RealTiCheck generates web search queries that can find relevant evidence to support or not support each claim. For each input claim, RealTiCheck uses an LLM to generate five web search queries. To prompt the LLM, RealTiCheck builds upon prior work that demonstrated the value of an "adversarial tone" in query generation for fact-checking [65], which encourages the model to generate questions that may be useful for debunking a claim. The prompt used is:

> You are a fact-checker looking for evidence to fact-check a particular claim. Your task is to generate search queries to find relevant information to support or not support the claim.
> Generate 5 different questions to criticize the following claim: {claim}
> If the given claim is not informative enough to generate a query, respond with "none." The generated questions should be diverse and cover distinct aspects of the claim.
> Ensure that each query is fully related to the claim and provides proper context. Also, explain the reasoning behind each query and its usefulness for fact-checking.

Step 2: Evidence Retrieval. Using each query, RealTiCheck collects evidence, or *documents*, through the web search; our implementation uses Google search. We chose Google as it is the search engine most commonly used by fact-checkers to identify evidence [37]. For each of the five generated queries per claim, RealTiCheck collects the top 10 ranked documents (i.e., the front page), producing a maximum of 50 unique documents per claim, following previous work [94]. RealTiCheck then de-duplicates documents retrieved through different queries and extracts the main text from each using either the `newspaper4k` Python library[4] for HTML documents or the `PyPDF2`[5] library for PDF documents.

[4]https://github.com/AndyTheFactory/newspaper4k
[5]https://pypi.org/project/PyPDF2/

Step 3: Chunking and Filtering Evidence. RealTiCheck processes retrieved evidence in two ways before entailment verification: (a) *chunking*, which determines how retrieved evidence is grouped; and (b) *filtering*, which removes potentially useless evidence. Prior work has taken varied approaches to both chunking and filtering: e.g., chunking evidence at a sentence-level [75], paragraph-level [28], or document-level [83]. In the absence of established best practices from prior research, RealTiCheck supports many different chunking and filtering strategies, which we evaluate further in Section 5.

Chunking. RealTiCheck applies three chunking strategies to each set of documents retrieved per claim, which we describe below:

(1) *Concatenated:* All documents retrieved for a single query are combined and then used to verify the claim.
(2) *Document:* Each document is used individually to verify the claim and the results are aggregated using a custom technique (discussed below).
(3) *Paragraph:* Each document is split into its constituent paragraphs on "\n\n," and each paragraph is used to verify the claim. The results are aggregated using the same custom technique.

Filtering. Web search returns a wide range of source types. To narrow this space, RealTiCheck provides two filtering knobs that can be applied independently or in combination: (a) a news filter, which restricts evidence to documents hosted on journalistic outlets, a proxy for the sources fact-checkers commonly draw on, and (b) a relevance filter, which removes chunks that are off-topic with respect to the claim. We evaluate the effect of each filter, alone and in combination, in Section 5.

For news filtering, RealTiCheck restricts evidence to documents whose hosting domain appears in the NewsHomepages project [87], a community-curated list of over 3,000 domains identified as news outlets. NewsHomepages is maintained by an open community of over 35 journalists, developers, and researchers, who add domains by submitting them to a public GitHub repository for review.

The list spans local, national, topic-specific, and international news outlets across the political spectrum, and inclusion reflects identification as a journalistic outlet rather than a credibility judgment. RealTiCheck checks each retrieved document's hosting domain against this list and removes documents whose domain is absent. The full list of news domains is available in our open-sourced code and data artifacts. We note that `newspaper4k`, used in Step 2 for HTML text extraction, plays no role in source filtering.

For relevance filtering, RealTiCheck uses an LLM (similar to prior work in an adjacent domain [27]) to determine whether an evidence chunk is related to the claim. We define "relevance" as follows:

Definition 4.1. Evidence is considered relevant if:

(1) It contains information directly related to the claim's subject matter.
(2) The information temporally aligns with the claim's context.
(3) It discusses the same entities or events mentioned in the claim.
(4) It provides substantive information about the claim's assertion.

We embed this definition to create an LLM-based relevance filter, the full prompt for which is on our github.

Step 4: Entailment Verification. After filtering evidence chunks, RealTiCheck then checks for "entailment"—a standard natural language processing task— in which a claim (i.e., hypothesis) is determined to be supported (or not) by evidence (i.e., premise). RealTiCheck uses an LLM to assess each claim-evidence pair, prompting it to return a verdict of "Supported" or "Not Supported", along with a brief justification that quotes directly from the evidence. The full prompt is:

> Claim: {claim}
> Evidence: {evidence}
> Given the claim and evidence, determine if the evidence supports or does not support the claim. First, provide a justification by citing a sentence or sentences from the premise to support your decision. Then, respond with:
> Answer: "Supported" or "Not Supported" only.

The output of this step is a {claim, verdict} pair for each retrieved evidence chunk.

Step 5: Aggregating Entailment Results. To determine the final verdict ("Supported" or "Not Supported") for a claim, RealTiCheck must aggregate the individual verdicts drawn from each evidence chunk. In prior research, aggregation is typically performed by a majority vote on the evidence [10]. However, a majority vote may be suboptimal in a fact-checking context—the burden of proof to support or not support a claim may be *higher* than a simple majority. Indeed, prior work has shown that fact-checkers collect a large body of evidence before making a final verdict [37].

To account for this, RealTiCheck uses a tunable, threshold-based approach to aggregation. The threshold specifies the percentage of agreement required among pieces of evidence before reaching a final verdict—e.g., a 70% "Not Supported" threshold means that at least 70% of the evidence collected about a claim must entail to "Not Supported" for the final verdict to be "Not Supported." We identify the most useful threshold for our use case in Section 5.3.

Step 6: Final Verdict and Evidence Chain. After aggregation, RealTiCheck produces a single, final verdict per claim, either "Supported" or "Not Supported," as well as an "evidence chain" that details the specific evidence that either supports or does not support the claim at hand. Although RealTiCheck is intended to support fact-checking efforts rather than replace human fact-checkers, we examine how RealTiCheck compares to the results of human fact-checkers in Sections 5 and 6.

5 Microbenchmarking RealTiCheck

In this section, we implement and evaluate RealTiCheck. This evaluation serves as a *benchmark* to examine how each RealTiCheck configuration presented in Section 4 impacts performance. For this evaluation, we utilize OpenAI's GPT-4o as our underlying LLM and Google search as our web retrieval engine. We observe little change in our results based on model-choice across frontier models. We provide a fuller evaluation with other models (Google Gemini, Anthropic Claude) in the Appendix 9. We selected GPT-4o as it performs best in all filtering configurations.

Dataset Used: AVeriTeC [64] is a recent dataset that has become a de-facto evaluation set in the NLP and security communities [47, 90, 93]. The dataset contains claim verification outcomes for claims sourced from previously fact-checked articles. The dataset comprises 4,568 real-world claims annotated with question-answer pairs generated from evidence and textual justifications, accompanied by metadata such as speaker, publisher, publication date, and relevant location. We used a random sample of 225 professionally fact-checked claims from AVeriTeC for our experiments, downsampling to balance both scale and costs for experimentation.

The 225 claims in the AVeriTeC dataset are grouped into four verdicts: 69 (30.67%) Supported, 116 (51.55%) Refuted, 23 (10.22%) Not Enough Evidence, and 17 (7.55%) Conflicting Evidence. For our task, we group refuted, conflicting evidence, and the not enough evidence verdicts into a broader "Not Supported" label. By doing this, our sample contains 156 (69.3%) "Not supported" claims and 69 (30.7%) "Supported" claims. To simulate a "real-time" component in our microbenchmark evaluations, we restrict evidence retrieval to only documents indexed within 10 days after the date each claim was originally published.

5.1 Evaluating Query Generation

To evaluate our query generation process, we compare our generated queries with the "questions" field available for each claim in the AVeriTeC dataset. These questions were generated by professional fact-checkers with the intention that the question might help veracity models through a Q&A reasoning task. For example, for the claim *"Hunter Biden had no experience in Ukraine or in the energy sector when he joined the board of Burisma"*, the questions that fact-checkers generated were 1. *"Did Hunter Biden have any experience in the energy sector at the time he joined the board of the Burisma energy company in 2014?"* and 2. *"Did Hunter Biden have any experience in Ukraine at the time he joined the board of the Burisma energy company in 2014?"*

We treat these questions as *proxies* for potential queries that a human labeler might enter into a search engine to verify a claim. For each claim in our sample of AVeriTeC, we generate search queries using the methodology described in Section 4 (**Step 1**), and compare our set of (generated) queries with AVeriTeC questions using the semantic similarity metrics ROUGE-1 [45], ROUGE-L [45], and BLEU [55]. As a baseline, we also compare these queries with a separate query generation method, which uses fine tuned models and prompts LLMs to generate literal and implied queries from claims in QuestGen [67]. We replicate their method by using their prompt and using GPT-4o as the backend model.

On ROUGE-1, ROUGE-L, and BLEU, our queries score slightly below QuestGen's on all three metrics, averaged over all generated queries per claim (0.31 vs. 0.37, 0.27 vs. 0.33, and 0.25 vs. 0.31, respectively), and neither approach exhibits substantial overlap with the fact-checker questions. This is consistent with previous work reporting the limitations of these metrics for evaluating generated queries [25, 46, 67]. We stress that this simply implies that properly evaluating query generation remains a challenging and fraught task, especially because queries phrased differently might still produce highly similar search results. We quantify the similarity in the search results by calculating the overlap in the domains returned by the queries generated by both methods. The median claim shares nine domains between the two methods, and a quarter of claims share 11 or more domains.

To evaluate the impact of query generation methods on the end result, we performed an experiment comparing QuestGen-generated queries with our query generation method through the full entailment pipeline on the same claims, documents, and relevance-filter configuration for the AVeriTeC dataset (identical to Section 5.4). RealTiCheck's queries achieve an M-F1 of 0.78 compared to 0.73 for QuestGen queries. This result suggests that despite the fact that our domains and queries differ lexically from QuestGen's, our strategy retrieves evidence that produces marginally better verification decisions.

5.2 Evaluating Data Division & Entailment

We evaluate how different chunking and filtering strategies applied to collected evidence impact claim verification performance, measuring how accurately our entailment-based predicted verdicts align with the ground truth provided in the AVeriTeC dataset. Performance is quantified using the macro F1 (M-F1) score, which accounts for both precision and recall across both verdict categories.

5.2.1 Evaluating Chunking. The chosen chunking strategy significantly impacts RealTiCheck's performance. In general, across all filtering configurations, *document-level* entailment performs better than both concatenated-chunking and paragraph-chunking, though the results are most stark when taking a single filter in isolation (i.e. news only or relevance only). For example, only applying relevance-filtering yields a document-chunk performance of 0.78 compared to a M-F1 of 0.67 for paragraph-chunking. This result suggests that current SOTA models can appropriately entail claims with a context length of a single document, but struggle when provided with either too much data, some of which may have contradictory opinions (i.e., concatenation) or too little data (i.e., paragraphs) which may not provide enough context to adjudicate the claim. This is corroborated by recent work [43], which showed a notable degradation in the reasoning performance of LLMs at input lengths much shorter than their technical maximum.

A notable observation is that RealTiCheck performs *slightly better* with document chunking than concatenated-chunking when "no filters" and "news only" filters are applied achieving a M-F1 of 0.66 and 0.72 respectively, also when "relevance only" is applied, where M-F1 scores for concatenated are 0.76 compared to 0.78 and only slightly worse when "all filters" are applied, where M-F1 score for concatenated is 0.82 and for document is 0.81 as shown in Table 1. This suggests that there is a threshold at which RealTiCheck can meaningfully process many different articles in a single context window and arrive at the correct verdict. This result is promising, as concatenated-chunking also reduces the number of queries made to the underlying LLM, which is the largest factor influencing RealTiCheck's runtime.

5.2.2 Evaluating Filtering. Across all chunking configurations, entailment performance improves with stricter filtering. For example, concatenated-chunking performance improves from 0.70 M-F1 to 0.82 M-F1 when all filters are applied, highlighting that enhancing the relevance and credibility of evidence is critical to improving automated fact-checking tooling.

Notably, we observe that *relevance* filtering is a much more effective filter than *news* filtering in practice. For example, in the document-chunking configuration, applying only a relevance filter improves M-F1 performance from 0.68 to 0.78, compared to an improvement to 0.72 with only news filtering, as shown in Table 1. This trend is consistent across all chunking configurations. Relevance filtering is also slightly less aggressive, reducing the percentage of verifiable claims to 64%–92% compared to 59%–60% with news filtering. Both filters in tandem work the best, but together reduce the verifiable claims to 42%–44%. In particular, news filtering removes an average of 7 (33.8%) documents per claim and 681 (78.3%) paragraphs per claim, and relevance filtering removes an average of 18 (85.7%) documents and 840 (96.5%) paragraphs per claim. We note that filtering also has the advantage of improving the time taken per claim, reducing the average time of completion from 21 minutes to 15.3 minutes for document chunking.

5.3 Evaluating Aggregation

Finally, we examine how aggregating by a threshold affects RealTiCheck performance. For document-chunking and paragraph-chunking, we explore the following question: *does majority voting always lead to maximal performance?* To measure this, we plot various F1 measurements (binary, weighted, micro F1, macro F1) across thresholds of "percentage of not supported votes required" from 0%–100% and observe where we achieve the highest performance. We show these thresholding curves with "relevance only" filters applied for document and paragraph level chunking in Figure 2.

Our results suggest that majority voting is indeed *insufficient* for optimal performance: for document-chunking, the optimal threshold to maximize M-F1 is 55%, slightly higher than a majority vote strategy. For paragraph-chunking, the optimal threshold is much higher, 85%, suggesting a much larger fraction of paragraphs must

	No Filter				News Only				Relevance Only				All Filters			
	M-P	M-R	M-F1	N	M-P	M-R	M-F1	N	M-P	M-R	M-F1	N	M-P	M-R	M-F1	N
Concat.	.70	.66	.68	225 (100%)	.70	.65	.66	135 (60%)	.76	.77	.76	142 (64%)	.84	.81	.82	94 (42%)
Document	.68	.69	.68	225 (100%)	.71	.74	.72	135 (60%)	.78	.78	.78	143 (64%)	.81	.83	.81	95 (42%)
Paragraphs	.66	.62	.62	225 (100%)	.72	.65	.66	132 (59%)	.68	.67	.67	208 (92%)	.75	.77	.75	99 (44%)

Table 1: Entailment Micro-Benchmarks— Macro Precision, Recall and F1 scores for all chunking and Filtering configurations.

	Maj. Vote				Optimal			
	P	R	F1	N	P	R	F1	N
Doc	.75	.75	.75	143 (64%)	.78	.78	.78	143 (64%)
Para	.71	.52	.45	206 (92%)	.68	.67	.67	208 (92%)

Table 2: Entailment Verification Performance on different Aggregation levels— Macro Precision, Recall, F1 scores, and sample count (N) for Majority Vote and for Optimal Threshold: 55% for Document (Doc) and 85% for Paragraph (Para)

Class	P	R	F1	N
Supported	.65	.67	.66	46
Refuted	.76	.73	.75	64
Not Enough Evidence	.29	.47	.36	17
Conflicting Evidence	.40	.12	.19	16
Macro Avg	.52	.50	.49	143

Table 3: Four-Class Entailment Performance—Per-class Precision, Recall, and F1 scores when preserving all four AVeriTec labels under the best binary configuration (document chunking, relevance-only filtering).

align before a claim verdict can be labeled as "Not Supported". Table 2 shows the comparisons for each chunking strategy in macro-precision, macro-recall, and macro-F1 when comparing the majority vote to an optimal threshold. We clearly see the jump in M-F1 performance for paragraph-chunking from 0.45 to 0.67 by changing the aggregation strategy from Majority Vote to Optimal Threshold (85%). Even for document-chunking, we observe a boost in performance by choosing the optimal threshold (55%) which has M-F1 of 0.78 instead of choosing the majority vote which has M-F1 of 0.75. We observe that dynamically tuning aggregation thresholds yields performance gains over the one-size-fits-all Majority Vote approach. This suggests that the aggregation strategy plays a significant role in automated fact-verification outcomes—a factor that has not been rigorously explored in prior work.

Importantly, across these configurations, RealTiCheck achieves up to 0.82 M-F1, substantially surpassing the prior state-of-the-art performance of 0.49 on the closest comparable tasks [65].

5.4 Overall Performance and Cost

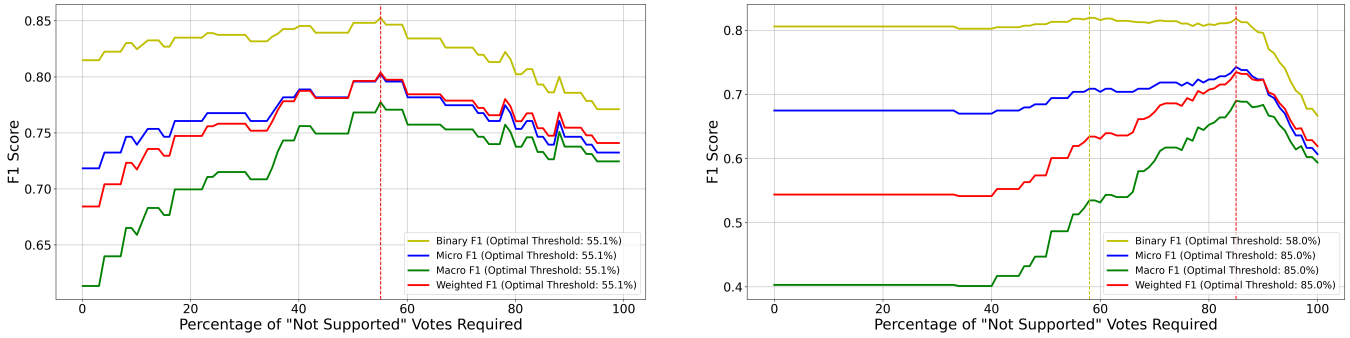
We utilize the macro-F1 score as our primary evaluation metric as fact-checking is evaluated on both positive and negative predictive power. Table 1 shows all results for all chunking and filtering configurations detailed in Section 4. With no filters applied (i.e. directly utilizing web evidence to entail claims), our system achieves an M-F1 of 0.62–0.68. Performance significantly improves when filters are added, with the "all filters" configuration achieving an M-F1 of 0.75–0.82. This result far exceeds prior work, which achieved M-F1s of 0.55–0.67 when only taking the "Supported" and "Not Supported" classes into account in AVeriTeC data [65].

While early research in claim entailment followed a four label scheme, our work follows recent papers that opt for binary choices of "Supported" and "Not Supported" which both improve model accuracy and better align with real world needs [11, 36, 89]. To evaluate this choice compared with a four-label configuration, we modified the entailment and aggregation steps of the pipeline. In the entailment step, we changed the prompt to return one of the four original AVeriTeC labels, *Supported*, *Refuted*, *Not Enough Evidence*, or *Conflicting Evidence/Cherrypicking*. In the aggregation step, we applied majority voting across the four labels per claim, breaking ties with a priority order (*Not Enough Evidence*, *Refuted*, *Conflicting Evidence/Cherrypicking*, *Supported*) consistent with AVeriTeC's label construction [65] and the AVeriTeC shared task [76]. We find a four-class problem yields an M-F1 of 0.49 compared to 0.78 under binary labeling while preserving the same RealTiCheck configuration (Table 3). Performance drop was primarily driven by the Not Enough Evidence (F1=0.36) and Conflicting Evidence (F1=0.19) classes, which together represent 23% of claims. Collapsing these labels into a "Not Supported" label enables existing fact-checking pipelines while also keeping RealTiCheck focused on triaging unsupported claims versus predicting exact labels.

The average time to verify a claim using RealTiCheck ranges from 14.75 minutes (concatenated chunking) to 45.61 minutes (paragraph chunking), depending on the chunking and filtering configuration. We do observe a tradeoff: as more available evidence is filtered, there are many cases where there is not sufficient evidence to properly entail the claim. When "all filters" are applied, for example, only 42–44% of claims returned a final verdict, compared to 64–92% when only the "relevance" filter is applied. We examine this tradeoff for recent claims in Section 6. End to end verification of 225 claims cost $117 in GPT-4o API fees ($0.52/claim), split roughly evenly between relevance filtering ($61) and entailment ($56). The input tokens (3,200 tokens/document on average) dominated the cost compared to output tokens.

6 RealTiCheck in Practice

To evaluate the effectiveness of RealTiCheck in a real world fact-checking scenario, we conducted a user study comparing RealTiCheck's fact-checking outcomes to those of expert fact-checkers on recent claims. By "recent," we mean claims that were published in the last 24 hours of being checked and require fact-checking with limited available evidence. Our evaluation seeks to approximate a "real" fact-checking process where claims are newer and evidence can be scant.



(a) Document-chunking **(b) Paragraph-chunking**

Figure 2: Thresholding Aggregation for Different Chunking Configurations— Binary F1, Weighted F1, Macro F1 and Micro F1 for different Threshold values. We show how selecting different threshold choices for aggregating entailment verdicts affects fact-checking performance. Our results show an optimal threshold of 55% and 85% for document and paragraph chunking, demonstrating that majority aggregation does not always lead to the best strategy in automated fact-checking with RealTiCheck.

We use Google Search as the retrieval engine throughout, following prior work that identified it as the engine most commonly used by fact-checkers [37]. To validate our search engine choice, ran the pipeline on the same 174 claims with Bing and find Google retrieves sufficient evidence for meaningfully more claims (170 vs. 139) and yields higher macro-F1 (0.71 vs. 0.64). We share the detailed analysis in Appendix 10.

In the study, these fact-checkers provide ground truth in determining whether claims are supported or not supported by evidence they find in their fact-checking process; this is typically evidence from online sources. To ensure a variety of claims, we ran our study over several days, sourcing new claims every day. In this section, we detail our experiment workflow and pipeline and describe how well RealTiCheck performed at fact-checking recent claims.

6.1 Study Workflow

We recruited fact-checkers to determine whether a set of provided claims were supported or not supported by online evidence. Below, we outline the process in more detail.

Recruitment: We recruited seven experienced fact-checkers via Upwork, a freelancing platform, to participate in our user study. All participants demonstrated relevant experience in fact-checking or expertise in closely related fields, such as journalism, politics, law, and public policy. To verify their expertise, we reviewed their Upwork profiles, which included portfolios and client feedback specifically relevant to fact-checking and analysis. Prior to the main study, we conducted a (successful) pilot study with one of the recruited fact-checkers to test the entire user study process. We ran our study over six days, from October 22nd, 2024 to October 28th, 2024, with fact-checking sessions held on five of those days. We scheduled 2 or 4 fact-checkers on each day. The study, including the pilot, was approved by our Institutional Review Board (IRB). Each fact-checker signed a digital informed-consent form before starting the task, and all work was performed remotely with no exposure to sensitive personal data. We paid each fact-checker $150 per session, each of which lasted on average 196.56 minutes (for an average hourly payment rate of $45.79).

Claim Selection: To align with RealTiCheck's goal of verifying *near real-time* information, we selected claims from recently published online content. To find such claims, we crawled articles from news domains identified by prior work [28, 29] as reliable or unreliable. Each day, we crawled 10 reliable domains (e.g., CNN, Wall Street Journal) and 10 unreliable domains (e.g., Daily Caller, Zero Hedge) and sourced articles that appeared in a 24 hour window.

We then used GPT-4o to identify sentences within each article that met the following definition of a "claim," drawn from prior work and fact-checker guidelines [88].

Definition 6.1. A claim is a sentence which is "check-worthy" and "factual". To elaborate, a sentence is a claim if it:

(1) Contains a factual assertion.
(2) Does not contain subjective sentences (e.g., opinions, beliefs, declarations).

We use this definition to prompt GPT-4o to identify claims. Further, we prompt GPT-4o to select the top three most "interesting" claims, where interestingness is based on general public interest and journalistic value. We reused the two criteria that emphasize the interest of the general public and journalists, as our goal was to identify claims that would capture the *greatest* attention from these groups. The other two criteria, which required claims to be factual and objective, were excluded at this stage. This decision was based on the fact that all sentences considered during this step had already been identified as claims by GPT-4o and, therefore, inherently satisfied the requirements of factuality and objectivity—criteria that are binary in nature. In contrast, the notion of "interestingness" is more nuanced and continuous. Here, our aim was to select the claims that not only attract attention from the general public and

Fact-checkers	d_1	d_2	d_3	d_4	d_6
FC1	×		×	×	
FC2	×	×			
FC3	×		×		
FC4	×		×		
FC5		×	×		×
FC6		×		×	
FC7		×			×

Table 4: Real-time Evaluation Schedule—Fact-checking sessions with seven fact-checkers from d_1 to d_6. On d_1–d_3, four fact-checkers were paired daily to evaluate the same 25 claims. On d_4, two fact-checkers evaluated another shared set of 25 claims. On d_6, each fact-checker individually evaluated 24 and 23 claims where previous pairs had disagreed.

journalists, but capture the *most* attention from them. We release full prompts for both these tasks in our github repository.

Claim Rephrasing & Filtering: We also prompted GPT-4o to rephrase each claim to remove ambiguities, such as unclear pronoun references, to ensure that each claim could be checked completely in isolation. In addition, this rephrasing step prevents fact-checkers from simply searching directly for the sentence in their online search process. Finally, we manually filtered to prioritize diversity in claims to ensure broad coverage of different news events. We selected claims spanning various topics and sourced from a range of websites, with no more than one claim chosen per article. This approach ensured a representative and varied set of claims for evaluation.

Claim Assignment: With our seven fact-checkers, we designed a schedule that spanned six days, with five active fact-checking days. We designed the schedule such that we always had two or four fact-checkers available on each day, such that each claim could be fact-checked by two people. The number of (finally) selected claims depended on the number of fact-checkers scheduled: 25 claims were selected if two fact-checkers were scheduled, and 50 claims if four fact-checkers were scheduled. Across all the five days with fact-checking sessions (one day was a break), 42.53% of the claims came from reliable sources, and 57.47% from unreliable sources. Table 4 shows our fact-checking schedule.

Interaction Protocol: Each day at 10 a.m. PST, we sent a Qualtrics form containing 25 claims to each participating fact-checker. Fact-checkers were instructed to label each claim as either "Supported" or "Not Supported" based on online evidence they could gather. For each claim, they were required to provide at least one URL reference, relevant source text, and a brief explanation of their judgment. For the study, we advised fact-checkers to remain politically neutral, avoid using fact-checking websites such as FactCheck.org as sources, and refrain from using large language models to generate responses or inform their decisions. Each claim was to be evaluated in a single, uninterrupted session, with fact-checkers recording the start and end times for each evaluation. To encourage accurate reporting of time, we specified in the Qualtrics form that their compensation would not be influenced by the duration they reported. All evaluations were required to be completed by 5 p.m. PST on the same day. Although we could not explicitly verify that each fact-checker followed all instructions, we found no evidence to the contrary in our final dataset. In sum, we collected ground truth fact-checked verdicts for a total of 175 claims. One claim was excluded due to accidental duplication, resulting in a final set of 174 unique claims. We present the claims we presented to fact-checkers in our GitHub repository.

	Supported	Not Supported
Reliable	67 (90.54%)	7 (9.46%)
Unreliable	62 (62%)	38 (38%)
Total	129	45

Table 5: Fact-checkers' Verdict Distribution by Source—Distribution of fact-checking verdicts for 174 claims, categorized by the reliability of their source domains ("Reliable" or "Unreliable"). This distribution highlights that claims from reliable domains were significantly more likely to be labeled as "supported" compared to those from unreliable domains.

Resolving Disagreements: Although fact-checkers mostly agreed on verdicts, there were 47 claims (27%) that fact-checkers disagreed on; these were claims where evidence was sparse and developing. To resolve disagreements, we recruited a third fact-checker who had previously never seen the claim before to provide an additional verdict and evidence to break the ties.

Deploying RealTiCheck in Real-Time with fact-checkers: In order to provide a true comparison to fact-checkers, we ran RealTiCheck on all claims assigned to fact-checkers every day on the same schedule as the fact-checkers. We did this to ensure parity between the evidence available to fact-checkers (i.e., through search or other means) and the evidence available to RealTiCheck. We also store all evidence retrieved from our search queries. As an additional step, RealTiCheck also filters out any documents sourced from third-party fact-checking domains (e.g., Politifact, Snopes) drawn from a list we compiled from the Duke Reporters' Lab's dataset of global fact-checking sites [61] to reduce potential bias as fact-checkers were instructed not to use existing fact-checked articles as evidence sources. We additionally discard any retrieved URL whose address contains fact-checking indicators (e.g., "fact-check," "factcheck," "truth-test," "fact-finders"), which catches fact-checking articles hosted on domains that are not themselves on the list.

6.2 Dataset

Table 5 presents a summary of fact-checking verdicts across 174 claims. Out of 74 claims sourced from reliable domains, 67 (90.54%) were labeled as "Supported," while 7 (9.46%) were labeled as "Not Supported." In contrast, for the 100 claims originating from unreliable domains, 62 (62%) were labeled as "Supported," and 38 (38%) were labeled as "Not Supported." This indicates that claims from reliable domains were significantly more likely to be labeled as "Supported" compared to those from unreliable domains. Overall, a total of 129 claims (74.14%) were found to be "Supported," and 45 claims (25.86%) were found to be "Not Supported" by the fact-checkers.

Evidence Domain Distribution: For the study, we instructed fact-checkers to provide at least one URL reference for each claim. For each claim, we examined the URLs cited by the two or three assigned fact-checkers and identified all unique domains (e.g., `www.cnn.com`, `www.nytimes.com`) referenced as sources. We then calculated the frequency with which each domain was cited across the 174 claims.

Domain	Count	Prop.	Domain	Count	Prop.
apnews.com	44	25.3%	thehill.com	14	8.0%
nytimes.com	40	23.0%	theguardian.com	13	7.5%
cnn.com	40	23.0%	npr.org	12	6.9%
nbcnews.com	24	13.8%	cnbc.com	10	5.7%
reuters.com	23	13.2%	bbc.com	9	5.2%
cbsnews.com	20	11.5%	usatoday.com	9	5.2%
newsweek.com	16	9.2%	abcnews.go.com	8	4.6%
x.com	15	8.6%	politico.com	7	4.0%
washingtonpost.com	15	8.6%	whitehouse.gov	7	4.0%
pbs.org	15	8.6%	axios.com	7	4.0%

Table 6: Top 20 Most Frequently Referenced Domains by Fact-Checkers—Domains most often cited across 174 claims. "Count" shows how many claims referenced each domain; "Prop." shows the percentage of total claims. Each domain was counted once per claim, regardless of how many fact-checkers cited it. These results highlight the key sources fact-checkers relied on, reflecting source credibility and diversity.

	Supported			Not Supported		
Threshold	P	R	F1	P	R	F1
25%	.84	**.83**	**.83**	.30	**.82**	.44
50%	.85	.63	.72	.34	.74	.47
75%	**.88**	.43	.58	.43	.53	.47
100%	.82	.18	.30	**.64**	.42	**.51**

Table 7: Performance by Supported vs. Not Supported Votes for all Claims. Precision (P), Recall (R), and F1 for all claims across varying vote agreement thresholds (25-100%). Lower thresholds favor "Supported" verdicts, while higher thresholds improve "Not Supported" classification, implying the trade-off between recall and precision.

Table 6 presents the 20 most frequently referenced domains, highlighting the credibility and diversity of sources utilized by fact-checkers during their fact-checking process.

6.3 Results

We evaluate RealTiCheck using 174 recent claims with fact-checker verdicts as ground truth.

Verification Performance: We begin by presenting the performance of RealTiCheck across the various evidence thresholds (Table 7). We show all configurations, rather than single "optimal" configuration (as in Section 5) to demonstrate the inherent tradeoffs RealTiCheck offers at different configuration levels. In particular, we present the binary precision, recall, and F1 scores for both classes when adjusting entailment thresholds for the percentage of "Supported" or "Not Supported" evidence verdicts in aggregate. For example, at a 25% support-evidence threshold, we see RealTiCheck achieves a support-class precision of 0.84 and support-class recall of 0.83.

In sum, we find that RealTiCheck is highly effective at identifying support-class claims and exhibits a standard precision-recall tradeoff with varying evidentiary standards. As the evidence threshold increases, we generally observe increased precision at the cost of recall—with the highest F1 score at 0.83 at a 25% threshold. Although precision improves somewhat with additional evidence (e.g., from 0.85 to 0.88 at a 75% threshold), this comes as a significant expense to recall (0.43). For "Not support" class claims, RealTiCheck achieves a maximum precision of 0.64 when *all evidence* fails to support the claim (i.e., 100% threshold) and a maximum recall of 0.82 at a 25% threshold.

	Supported			Not Supported		
Threshold	P	R	F1	P	R	F1
25%	.87	**.88**	**.87**	.56	.54	.55
50%	**.92**	.68	.78	.42	**.78**	.54
75%	**.97**	.34	.50	.30	**.97**	.45
100%	**1.00**	.02	.04	.23	**1.00**	.37

Table 8: Performance by Supported vs. Not Supported Votes for Unanimous Claims. We show performance for claims where fact-checkers had initial agreement (i.e., excluding the claims on which they disagreed and a third fact-checker was recruited to break ties). On claims where expert fact-checkers arrive at a unanimous verdict, RealTiCheck's performance improves across thresholds, achieving a maximum support-class F1 of 0.87 at a threshold of 25%.

Interestingly, we find that RealTiCheck performance improves further and nearly perfectly aligns with fact-checker verdicts when the human fact-checkers initially agree on the claim verdict. Table 8 shows both "Supported" class and "Not Supported" class performance metrics for the 127 claims where both fact-checkers initially agreed. Performance increases significantly for support-class metrics, with a maximum support-class F1 of 0.87 at a 25% evidence threshold. At the optimal threshold that maximizes macro-F1, we find that RealTiCheck achieves 91% accuracy with macro-precision of 0.82, macro-recall of 0.74, and macro-F1 of 0.77.

These findings offer important implications for deploying RealTiCheck in real-world contexts. Compared to RealTiCheck performance on AVeriTeC claims (Section 5), which are highly curated pieces of more obvious falsehoods, our collection of recent claims from a wide mix of trustworthy and untrustworthy sources presents a significantly harder problem for automatic detection and entailment. Even in this case, RealTiCheck performs well on support-class claims, achieving an F1 of 0.83 at a 25% evidence threshold. A potential opportunity for using RealTiCheck in practice could be to prioritize review of claims that are labeled as "Not Supported" at this threshold, as these are much less likely to be supported by available evidence. RealTiCheck's tunable thresholds, however, enables fact-checking organizations to further adapt the system to their specific needs and review capacity, optimizing either for identifying claims that are supported by evidence (low thresholds on "Supported"), minimizing false accusations (high thresholds on "Not Supported"), or balancing both objectives simultaneously.

Exploring misclassifications: For the configuration: "relevance only" and document-chunking, which has the best performance without losing a lot of claims, we found 41 claims were misclassified out of N=170 claims in that configuration, out of which 13 were false positives (a claim is flagged as "Supported" when it is "Not Supported") and 28 were false negatives (a claim was flagged as "Not supported" when the claim was "Supported"). Using iterative

Tag	%
Partial Evidence	56%
Numerical Comparison	18%
Non useful Evidence	14%
Direct Contradiction	9%
Model Mistake	3%

(a) False neg.

Tag	%
Partial Evidence	74%
Direct Contradiction	20%
Numerical Comparison	6%
–	
–	

(b) False pos.

Table 9: Explaining errors in Entailment Verification on Real-Time Data—We show the most common explanations for errors generated by RealTiCheck. False positives and false negatives are largely due to partial evidence, where RealTiCheck overgeneralizes towards an outcome. Direct numerical comparisons are prone to be classified as "Not Supported" defacto. On the other hand, direct contradictions lead to incorrect "Supported" outcomes.

open-coding, we labeled a sample of four incorrect entailments per claim and study why models make such errors (Table 9).

For false positives, available evidence can confirm part of the claim (but not in entirety), however, the per-chunk "Supported" rate is high enough to cross threshold even though the model overlooks key qualifiers. For example, consider the claim: "Andrew Yang proposed that every person in the United States would receive $1,000 per month as part of his universal basic income plan," for which all 26 retrieved evidence chunks (100%) returned "Supported." Each chunk confirmed Yang's UBI proposal of $1,000 per month, but Yang's actual "Freedom Dividend" targeted "every American adult," not "every person." A similar pattern appears in the claim: *"Starting in 2026, the federal government will negotiate the prices of 10 of the most expensive drugs covered by Medicare for the first time."* The model retrieved the evidence snippet: *"On August 29, 2023, CMS published the list of 10 drugs covered under Medicare Part D selected for the first cycle of negotiation,"* and concluded the claim was supported. However, this evidence merely mentions the selection of drugs for negotiation and verifies neither the central assertion about price negotiation nor the 2026 timeline.

False negatives occur in compound claims where evidence corroborates each individual component of the claim but rarely all of them together in a single chunk, so the share of "Supported" verdicts falls below RealTiCheck's aggregation threshold. For example, in the claim "Michigan has increased security at election centers statewide after Trump supporters attempted to storm a counting center in Detroit during the 2020 election," retrieved evidence that confirmed both the 2020 storming attempt and recent statewide security enhancements, but most chunks discussed only one component or the other, and few established the temporal-causal link between the two.

In both cases, a unifying issue is that the model reasons over partial evidence which is a well-documented LLM limitation [20], rather than tracking which components of a claim each chunk does and does not verify. Treating "Supported" as the positive class for this analysis, the error distribution in Table 9 shows that partial-evidence errors account for 56% of false negatives (compound-claim cases like the Michigan example, where evidence is scattered across chunks) and 74% of false positives (overgeneralization cases like the Yang and Medicare examples, where partial confirmation gets extrapolated to full support). The rest of the errors are split across numerical comparisons, direct contradictions, and other model mistakes.

RealTiCheck also encounters challenges with claims involving direct and specific numerical comparisons—a well-documented limitation across LLM-based fact-checking systems [4]. In the absence of an exact value match, automated systems tend to default to a "Not Supported" verdict, whereas human fact-checkers can leverage contextual reasoning to interpret approximate matches and assess the overall validity of a claim. This represents an important area for future development in deploying LLMs for numeric-heavy fact-checking settings, where the interpretation of quantitative evidence often requires nuanced judgment beyond exact matching.

7 Discussion and Conclusion

In this section, we discuss the design implications and considerations of the use for RealTiCheck, and present some concrete directions for future research in automated fact-checking.

7.1 What Should LLMs be Responsible For?

Recent developments in research and industry aim to embed both web search and reasoning directly into LLMs. Systems like Perplexity, AskGrok, and newer OpenAI models integrate retrieval and generation, often as black boxes with limited interpretability and potential biases [14, 77]. Although LLMs getting access to tools such as web search allow various applications, it has been shown that this can lead to malicious use [40].

LLMs have been shown to be effective for aiding trust and safety tasks such as content moderation [73] in assisting human raters where they assist rather than replace human raters. Our central design choice is to restrict the LLM's role to natural-language reasoning, while evidence collection, filtering, and aggregation remain configurable based on fact-checker needs. By decoupling these stages, RealTiCheck supports explicit policy enforcement (e.g., source allowlists, query rate limits) and reproducible verification trails, making sure evidence traceability and information fidelity is not compromised. In tandem, we observe this limited scope also reduces the surface for hallucinations: To quantify this, we conducted an experiment measuring hallucination rates in RealTiCheck's entailment and relevance verdicts. Specifically, we randomly sampled 100 claim–evidence pairs from our evaluation set and manually annotated them for hallucinations, defined as instances where the model introduced content not supported by the provided evidence. We implemented this check as a lightweight annotation task, with the lead author reviewing each verdict–evidence pair. In this sample, we did not identify any hallucinations. While fallacies remain possible when relying on LLMs, our findings suggest they are rare in this restricted reasoning-only setting.

7.2 Considerations of Use

There are several boundaries on acceptable use-cases for tools like RealTiCheck in practice:

Interpreting Verdicts. RealTiCheck is best applied to claims that can be checked against publicly available, published evidence. Furthermore, we stress that while the system provides a verdict for a claim, that we do not seek to decide whether a claim is true or false, but rather, whether the existing claim is supported by the

collected evidence. In security-relevant domains such as misinformation during elections, early reports are often sparse, fragmented, or deliberately seeded with false leads. In such cases, the system's "Not Supported" verdict should be interpreted as evidence absent, not evidence of falsehood—a distinction crucial for avoiding suppression of true but underreported information.

Triage Policy for Fact-checkers: The goal of our system is to assist human fact-checkers in reducing fact-checking load and not act as a replacement for them. To achieve this, we leverage the high precision for the "Supported" class to reduce the load for fact-checkers. This nature of triage has also been adopted in other contexts such as content moderation and has successfully reduced the reviewer load [73]. Although every claim ultimately requires fact-checker adjudication, this triage approach enables fact-checkers to prioritize their efforts rather than treating all claims equally. By surfacing likely problematic claims first through the "Not Supported" queue, fact-checkers can focus their initial attention on high-impact cases where intervention is most needed, improving the efficiency and effectiveness of the overall fact-checking workflow.

Triage by claim-type. Not all claims have equivalent evidence wealth and reporting, leaving the potential for some *types* of claims to be more readily analyzed than others. In a preliminary experiment, we categorized 91 claims from the AVeriTeC data sample into three groups: Health & Science (36 claims), Politics (40 claims), and Economics (15 claims), and found that each had vastly different success rates (0.86, 0.79, 0.60 M-F1 respectively). As such, future work might explore early detection of the type of claim and use this information to triage automated and human-effort.

Evidence Selection. Evidence selection can dramatically impact RealTiCheck's performance. We currently leverage web search (Google) and a filter restricting evidence to identified news outlets via NewsHomepages [87]. We found that restricting the evidence to news sources only marginally improved verification performance. For instance, using only mainstream news articles as evidence yielded about a 5.9% increase in macro-F1, filtering for relevant documents gave a 14.7% boost and applying both filters together achieved a 19.1% improvement. These gains highlight that filtering for relevance and news sources both contribute to verification performance, with relevance filtering producing the larger effect.

Temporality of Outcomes. RealTiCheck's verdict is bounded by the available evidence during fact-checking. RealTiCheck adjudicates against evidence available at query time, and as that evidence evolves, the correct verdict for the same claim can change. In rapidly evolving situations such as natural disasters, early reports are sparse and frequently revised. For example, during the August 2023 Maui wildfires, claims about which roads were closed and which areas were under mandatory evacuation shifted hourly as the fire advanced and conditions changed [35]. A claim such as "the Lahaina Bypass is closed to civilian traffic" would have flipped multiple times within a single day, and RealTiCheck verdicts on such claims would have shifted accordingly. Claims can also *start false and become true* as conditions catch up to the assertion. For example, the claim "eight million people are enrolled in President Biden's SAVE plan" was inaccurate in September 2023 when about four million had enrolled [72], but became accurate by early 2024 once enrollment crossed the threshold [53]. The reverse holds as well, that is, a claim that started as true can become false. For example, "Kevin McCarthy is the Speaker of the House" was correct from January through October 2023, when he became the first Speaker removed by motion to vacate [71]. When interpreting a RealTiCheck verdict, users should treat it as a snapshot tied to the time it was checked rather than a standing label of veracity.

7.3 Limitations

Our work has several limitations. First, the study was conducted in the US and thus was only evaluated using claims in English and evidence based in the US. Furthermore, RealTiCheck currently uses *only* textual data as evidence: as such, it cannot immediately verify claims tied to images, videos, or other multimedia content. While we sought to emulate a more realistic deployment scenario, more work is necessary to evaluate how a tool like this might be implemented in tandem with fact-checkers and evaluated in the context of their workplace or work environment. RealTiCheck is also downstream of several systems, such as search and commercial LLMs, and as such, changes to these systems may also impact RealTiCheck's performance.

7.4 Future Extensions of RealTiCheck

Implementing RealTiCheck for End-Users. A natural next step in the implementation of RealTiCheck is to design interfaces that seamlessly integrate fact-checking into everyday information consumption. One promising avenue is the development of a browser extension that allows users to highlight or select claims encountered online and receive, in real time, both a system verdict and an accompanying explanation. This interface enables visibility into the evidence chain by displaying retrieved documents, intermediate reasoning, and citations alongside the final verdict. This transparency could increase user trust and support critical engagement by showing not only what the system concluded, but also why it reached that conclusion. Integrating RealTiCheck in this way stresses human–AI collaboration: rather than replacing user judgment, the system can serve as an aid that provides structured evidence and rationales, leaving the ultimate decision-making to the individual. Design choices should emphasize usability and interpretability, such as interactive evidence trails and collapsible justification panels. These considerations illustrate how RealTiCheck could be translated from research to practice in settings where rapid and reliable credibility assessments are most needed, such as during elections or public health crises.

8 Acknowledgments

We thank the professional fact-checkers for their contributions to this work, as well as Varun Chandrasekaran, Stefan Savage, and Geoff Voelker, who helped to shape and clarify the contributions of this work along the way.

References

[1] 2025. PolitiFact. https://www.politifact.com/.
[2] 2025. Snopes: The definitive fact-checking site and reference source for urban legends, folklore, myths, rumors, and misinformation. https://www.snopes.com/
[3] Bill Adair, Chengkai Li, Jun Yang, and Cong Yu. 2017. Progress toward "the holy grail": The continued quest to automate fact-checking. In *Computation+Journalism Symposium.*

	No Filter				News Only				Relevance Only				All Filters			
	M-P	M-R	M-F1	N	M-P	M-R	M-F1	N	M-P	M-R	M-F1	N	M-P	M-R	M-F1	N
Gemini 2.0 Flash	.62	.63	.62	225 (100%)	.59	.59	.59	200 (88%)	.64	.60	.60	181 (80%)	.69	.72	.68	123 (55%)
GPT-4o	.68	.69	.68	225 (100%)	.71	.74	.72	135 (60%)	.78	.78	.78	143 (64%)	.81	.83	.81	95 (42%)
Claude 3.5 Sonnet	.70	.67	.68	225 (100%)	.72	.64	.65	200 (88%)	.71	.74	.72	169 (75%)	.76	.78	.77	111 (49%)

Table 10: Entailment Results - Model Choice— Macro Precision, Recall and F1 scores for different model choices for all filtering configurations and document chunking.

[4] Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large Language Models for Mathematical Reasoning: Progresses and Challenges. In *Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*.

[5] Jennifer Allen, Antonio A Arechar, Gordon Pennycook, and David G Rand. 2021. Scaling up fact-checking using the wisdom of crowds. *Science advances* 7, 36 (2021), eabf4393.

[6] Jennifer Allen, Cameron Martel, and David G Rand. 2022. Birds of a feather don't fact-check each other: Partisanship and the evaluation of news in Twitter's Birdwatch crowdsourced fact-checking program. In *Proceedings of the 2022 CHI conference on human factors in computing systems*. 1–19.

[7] Allison Baker and Viviane Fairbank. 2022. A. Reporter's Guidelines for Fact-Checked Journalism. https://thetijproject.ca/guide/reporter-guidelines/

[8] Nadav Borenstein, Greta Warren, Desmond Elliott, and Isabelle Augenstein. 2025. Can community notes replace professional fact-checkers?. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 535–552.

[9] Wilson Ceron, Mathias-Felipe de Lima-Santos, and Marcos G. Quiles. 2021. Fake news agenda in the era of COVID-19: Identifying trends through fact-checking content. *Online Social Networks and Media* (2021).

[10] Jifan Chen, Grace Kim, Aniruddh Sriram, Greg Durrett, and Eunsol Choi. 2024. Complex Claim Verification with Evidence Retrieved in the Wild. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[11] Yingjian Chen, Haoran Liu, Yinhong Liu, Jinxiang Xie, Rui Yang, Han Yuan, Yanran Fu, Peng Yuan Zhou, Qingyu Chen, James Caverlee, et al. 2025. GraphCheck: Breaking long-term text barriers with extracted knowledge graph-powered fact-checking. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

[12] Eun Cheol Choi and Emilio Ferrara. 2024. Fact-gpt: Fact-checking augmentation via claim matching with llms. In *Companion Proceedings of the ACM Web Conference*.

[13] Yuwei Chuai, Haoye Tian, Nicolas Pröllochs, and Gabriele Lenzini. 2024. Did the roll-out of community notes reduce engagement with misinformation on X/Twitter? *Proceedings of the ACM on human-computer interaction* 8, CSCW2 (2024), 1–52.

[14] Sunhao Dai, Chen Xu, Shicheng Xu, Liang Pang, Zhenhua Dong, and Jun Xu. 2024. Bias and unfairness in information retrieval systems: New challenges in the llm era. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 6437–6447.

[15] Anubrata Das, Houjiang Liu, Venelin Kovatchev, and Matthew Lease. 2023. The state of human-centered NLP technology for fact-checking. (2023).

[16] Laurence Dierickx, Carl-Gustav Lindén, and Andreas Lothe Opdahl. 2023. Automated fact-checking to support professional practices: systematic literature review and meta-analysis. *International Journal of Communication* (2023).

[17] Tim Draws, David La Barbera, Michael Soprano, Kevin Roitero, Davide Ceolin, Alessandro Checco, and Stefano Mizzaro. 2022. The Effects of Crowd Worker Biases in Fact-Checking Tasks. In *ACM Conference on Fairness, Accountability, and Transparency*.

[18] Tamer Elsayed, Preslav Nakov, Alberto Barrón-Cedeño, Maram Hasanain, Reem Suwaileh, Giovanni Da San Martino, and Pepa Atanasova. 2019. Overview of the CLEF-2019 CheckThat! Lab: Automatic Identification and Verification of Claims. In *European Conference on IR Research (ECIR)*.

[19] FactCheck.org. 2025. FactCheck.org: Our Process. https://www.factcheck.org/our-process/

[20] Huawen Feng, Yan Fan, Xiong Liu, Ting-En Lin, Zekun Yao, Yuchuan Wu, Fei Huang, Yongbin Li, and Qianli Ma. 2024. Improving Factual Consistency of News Summarization by Contrastive Preference Optimization. In *Findings of the Association for Computational Linguistics: EMNLP*.

[21] Full Fact. 2025. Full Fact: Frequently Asked Questions. https://live.fullfact.org/about/frequently-asked-questions/

[22] Max Glockner, Yufang Hou, and Iryna Gurevych. 2022. Missing Counter-Evidence Renders NLP Fact-Checking Unrealistic for Misinformation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[23] Lucas Graves. 2018. *Understanding the Promise and Limits of Automated Fact-Checking*. Technical Report. Reuters Institute for the Study of Journalism.

[24] Lucas Graves and Michelle A. Amazeen. 2019. Fact-Checking as Idea and Practice in Journalism. Oxford University Press.

[25] Max Grusky. 2023. Rogue scores. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

[26] Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics* (2022).

[27] Catherine Han, Anne Li, Deepak Kumar, and Zakir Durumeric. 2024. PressProtect: Helping Journalists Navigate Social Media in the Face of Online Harassment. *Proceedings of the ACM on Human-Computer Interaction* (2024).

[28] Hans WA Hanley, Deepak Kumar, and Zakir Durumeric. 2024. Specious sites: Tracking the spread and sway of spurious news stories at scale. In *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1609–1627.

[29] Hans W. A. Hanley and Zakir Durumeric. 2024. Machine-Made Media: Monitoring the Mobilization of Machine-Generated Articles on Misinformation and Mainstream News Websites. In *International AAAI Conference on Web and Social Media (ICWSM)*.

[30] Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-Athene: Multi-Sentence Textual Entailment for Claim Verification. In *Fact Extraction and VERification Workshop (FEVER)*.

[31] Naeemul Hassan, Bill Adair, James T. Hamilton, Chengkai Li, Mark Tremayne, Jun Yang, and Cong Yu. 2015. The quest to automate fact-checking. In *Computation+Journalism Symposium*.

[32] Naeemul Hassan, Fatma Arslan, Chengkai Li, and Mark Tremayne. 2017. Toward Automated Fact-Checking: Detecting Check-Worthy Factual Claims by ClaimBuster. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

[33] Naeemul Hassan, Chengkai Li, and Mark Tremayne. 2015. Detecting Check-worthy Factual Claims in Presidential Debates. In *ACM International Conference on Information and Knowledge Management*.

[34] Naeemul Hassan, Gensheng Zhang, Fatma Arslan, Josue Caraballo, Damian Jimenez, Siddhant Gawsane, Shohedul Hasan, Minumol Joseph, Aaditya Kulkarni, Anil Kumar Nayak, Vikas Sable, Chengkai Li, and Mark Tremayne. 2017. ClaimBuster: the first-ever end-to-end fact-checking system. In *VLDB Endowment International Conference on Very Large Data Bases*.

[35] Honolulu Civil Beat. 2023. Live Updates: Maui Fire Evacuations, Closures and Shelter Updates. https://www.civilbeat.org/2023/08/live-maui-fire-evacuations-closures-and-shelter-updates/. Accessed: 2026-05.

[36] Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. 2020. HoVer: A dataset for many-hop fact extraction and claim verification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

[37] Prerna Juneja and Tanushree Mitra. 2022. Human and Technological Infrastructures of Fact-Checking. *Proceedings of the ACM on Human-Computer Interaction* (2022).

[38] Payam Karisani and Heng Ji. 2024. Fact Checking Beyond Training Set. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[39] Kashif Khan, Ruizhe Wang, and Pascal Poupart. 2022. WatClaimCheck: A new Dataset for Claim Entailment and Inference. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

[40] Hanna Kim, Minkyoo Song, Seung Ho Na, Seungwon Shin, and Kimin Lee. 2025. When LLMs Go Online: The Emerging Threat of Web-Enabled LLMs. In *Proceedings of the 34th USENIX Security Symposium (USENIX Security 25)*. USENIX Association.

[41] Neema Kotonya and Francesca Toni. 2020. Explainable Automated Fact-Checking: A Survey. In *International Conference on Computational Linguistics (COLING)*.

[42] Sian Lee, Aiping Xiong, Haeseung Seo, and Dongwon Lee. 2023. "Fact-checking" fact checkers: A data-driven approach. *Harvard Kennedy School Misinformation Review* (2023).

[43] Mosh Levy, Alon Jacoby, and Yoav Goldberg. 2024. Same Task, More Tokens: the Impact of Input Length on the Reasoning Performance of Large Language Models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

[44] Miaoran Li, Baolin Peng, Michel Galley, Jianfeng Gao, and Zhu Zhang. 2024. Self-checker: Plug-and-play modules for fact-checking with large language models. In *Findings of the association for computational linguistics: NAACL 2024*. 163–181.

[45] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*.

[46] Feifan Liu and Liu Yang. 2010. Exploring Correlation Between ROUGE and Human Evaluation on Meeting Summaries. *IEEE Transactions on Audio, Speech, and Language Processing* (2010).

[47] Jin Liu, Steffen Thoma, and Achim Rettinger. 2024. FZI-WIM at AVeriTeC Shared Task: Real-World Fact-Checking with Question Answering. In *Fact Extraction and VERification Workshop (FEVER)*.

[48] Weiyao Luo, Junfeng Ran, Zailong Tian, Sujian Li, and Zhifang Sui. 2024. FaGANet: An Evidence-Based Fact-Checking Model with Integrated Encoder Leveraging Contextual Information. In *Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING)*.

[49] Cameron Martel and David G Rand. 2024. Fact-checker warning labels are effective even for those who distrust fact-checkers. *Nature Human Behaviour* 8, 10 (2024), 1957–1967.

[50] Nicholas Micallef, Vivienne Armacost, Nasir Memon, and Sameer Patil. 2022. True or false: Studying the work practices of professional fact-checkers. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW1 (2022), 1–44.

[51] Preslav Nakov, Alberto Barrón-Cedeño, Tamer Elsayed, Reem Suwaileh, Lluís Màrquez, Maram Hasanain, Firoj Alam, and Fatima Haouari. 2018. Overview of the CLEF-2018 CheckThat! Lab on Automatic Identification and Verification of Political Claims. In *European Conference on IR Research (ECIR)*.

[52] Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021. Automated fact-checking for assisting human fact-checkers. *arXiv preprint arXiv:2103.07769* (2021).

[53] National Credit Union Administration. 2024. Save with the SAVE Plan. https://mycreditunion.gov/about/news-blog/save-save-plan. Accessed: 2026-05.

[54] Rrubaa Panchendrarajan and Arkaitz Zubiaga. 2024. Claim detection for automated fact-checking: A survey on monolingual, multilingual and cross-lingual research. *Natural Language Processing Journal* (2024).

[55] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics*.

[56] PolitiFact. 2018. The Principles of the Truth-O-Meter: PolitiFact's Methodology. https://www.politifact.com/article/2018/feb/12/principles-truth-o-meter-politifacts-methodology-i/

[57] Ethan Porter and Thomas J Wood. 2021. The global effectiveness of fact-checking: Evidence from simultaneous experiments in Argentina, Nigeria, South Africa, and the United Kingdom. *Proceedings of the National Academy of Sciences* 118, 37 (2021), e2104235118.

[58] Nicolas Pröllochs. 2022. Community-based fact-checking on Twitter's Birdwatch platform. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 16. 794–805.

[59] Dorian Quelle and Alexandre Bovet. 2024. The perils and promises of fact-checking with large language models. *Frontiers in Artificial Intelligence* (2024).

[60] Thomas Renault, David Restrepo-Amariles, and Aurore Troussel. 2024. Collaboratively adding context to social media posts reduces the sharing of false news. *Available at SSRN 4800565* (2024).

[61] Reporters' Lab. 2025. Fact-Checking. https://reporterslab.org/fact-checking/

[62] Mohammed Saeed, Nicolas Traub, Maelle Nicolas, Gianluca Demartini, and Paolo Papotti. 2022. Crowdsourced fact-checking at Twitter: How does the crowd compare with experts?. In *ACM International Conference on Information and Knowledge Management*.

[63] Mourad Sarrouti, Asma Ben Abacha, Yassine Mrabet, and Dina Demner-Fushman. 2021. Evidence-Based Fact-Checking of Health-Related Claims. In *Findings of the Association for Computational Linguistics: EMNLP*.

[64] Michael Schlichtkrull, Yulong Chen, Chenxi Whitehouse, Zhenyun Deng, Mubashara Akhtar, Rami Aly, Zhijiang Guo, Christos Christodoulopoulos, Oana Cocarascu, Arpit Mittal, et al. 2024. The Automated Verification of Textual Claims (AVeriTeC) Shared Task. In *Fact Extraction and VERification Workshop (FEVER)*.

[65] Michael Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. 2024. Averitec: A dataset for real-world claim verification with evidence from the web. *Advances in Neural Information Processing Systems* (2024).

[66] Connie Moon Sehat, Ryan Li, Peipei Nie, Tarunima Prabhakar, and Amy X Zhang. 2024. Misinformation as a harm: structured approaches for fact-checking prioritization. *Proceedings of the ACM on Human-Computer Interaction* (2024).

[67] Ritvik Setty and Vinay Setty. 2024. QuestGen: Effectiveness of Question Generation Methods for Fact-Checking Applications. In *ACM International Conference on Information and Knowledge Management*.

[68] Craig Silverman. 2025. Verification and Fact Checking. https://datajournalism.com/read/handbook/verification-1/additional-materials/verification-and-fact-checking

[69] V. Singrodia, A. Mitra, and S. Paul. 2019. A review on web scrapping and its applications. In *2019 International Conference on Computer Communication and Informatics (ICCCI)*. IEEE, 1–6.

[70] Isaac Slaughter, Axel Peytavin, Johan Ugander, and Martin Saveski. 2025. Community notes reduce engagement with and diffusion of false information online. *Proceedings of the National Academy of Sciences* 122, 38 (2025), e2503413122.

[71] The Washington Post. 2023. Kevin McCarthy removed as House Speaker in unprecedented vote. https://www.washingtonpost.com/politics/2023/10/03/kevin-mccarthy-house-speaker-vote-motion-to-vacate/. Accessed: 2026-05.

[72] The White House. 2023. What They Are Reading in the States: More than 4 Million Student Loan Borrowers Enrolled in New Biden-Harris Administration SAVE Plan. https://bidenwhitehouse.archives.gov/briefing-room/statements-releases/2023/09/09/what-they-are-reading-in-the-states-more-than-4-million-student-loan-borrowers-enrolled-in-new-biden-harris-administration-save-plan/. Accessed: 2026-05.

[73] Kurt Thomas, Patrick Gage Kelley, David Tao, Sarah Meiklejohn, Owen Vallis, Shunwen Tan, Blaž Bratanič, Felipe Tiengo Ferreira, Vijay Kumar Eranti, and Elie Bursztein. 2025. Supporting Human Raters with the Detection of Harmful Content using Large Language Models. In *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2772–2789.

[74] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a Large-scale Dataset for Fact Extraction and Verification. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[75] James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018. The Fact Extraction and VERification (FEVER) Shared Task. In *Workshop on Fact Extraction and VERification (FEVER)*.

[76] Herbert Ullrich, Tomáš Mlynář, and Jan Drchal. 2024. AIC CTU system at AVeriTeC: Re-framing automated fact-checking as a simple RAG task. In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*. 137–150.

[77] Pranav Narayanan Venkit, Philippe Laban, Yilun Zhou, Yixin Mao, and Chien-Sheng Wu. 2024. Search Engines in an AI Era: The False Promise of Factual and Verifiable Source-Cited Responses. *arXiv preprint arXiv:2410.22349* (2024).

[78] Andreas Vlachos and Sebastian Riedel. 2014. Fact checking: Task definition and dataset construction. In *ACL Workshop on Language Technologies and Computational Social Science*.

[79] Juraj Vladika and Florian Matthes. 2023. Scientific Fact-Checking: A Survey of Resources and Approaches. In *Findings of the Association for Computational Linguistics: ACL*.

[80] Soroush Vosoughi, Deb Roy, and Sinan Aral. 2018. The spread of true and false news online. *Science* (2018).

[81] Ivan Vykopal, Matúš Pikuliak, Simon Ostermann, and Marián Šimko. 2024. Generative Large Language Models in Automated Fact-Checking: A Survey. https://arxiv.org/abs/2407.02351

[82] Morgan Wack, Kayla Duskin, and Damian Hodel. 2026. Political fact-checking efforts are constrained by deficiencies in coverage, speed, and reach. *Political Communication* (2026), 1–24.

[83] David Wadden, Kyle Lo, Lucy Lu Wang, Arman Cohan, Iz Beltagy, and Hannaneh Hajishirzi. 2022. MultiVerS: Improving scientific claim verification with weak supervision and full-document context. In *Findings of the Association for Computational Linguistics: NAACL*.

[84] Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, Yidong Wang, Linyi Yang, Jindong Wang, Xing Xie, Zheng Zhang, and Yue Zhang. 2023. Survey on Factuality in Large Language Models: Knowledge, Retrieval and Domain-Specificity. https://arxiv.org/abs/2310.07521

[85] Haoran Wang and Kai Shu. 2023. Explainable claim verification via knowledge-grounded reasoning with large language models. In *Findings of the association for computational linguistics: EMNLP 2023*. 6288–6304.

[86] Greta Warren, Irina Shklovski, and Isabelle Augenstein. 2025. Show me the work: Fact-checkers' requirements for explainable automated fact-checking. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–21.

[87] Ben Welsh, Naitian Zhou, Arda Kaz, Michael Vu, and Alexander Spangher. 2024. NewsHomepages: Homepage Layouts Capture Information Prioritization Decisions. *arXiv preprint arXiv:2501.00004* (2024).

[88] Madelyne Xiao and Jonathan Mayer. 2025. SoK: Machine Learning for Misinformation Detection. In *Proceedings of the 34th USENIX Security Symposium*. USENIX Association, Seattle, WA, USA. https://www.usenix.org/conference/usenixsecurity25/presentation/xiao-madelyne

[89] Zhuohan Xie, Rui Xing, Yuxia Wang, Jiahui Geng, Hasan Iqbal, Dhruv Sahnan, Iryna Gurevych, and Preslav Nakov. 2025. FIRE: fact-checking with iterative retrieval and verification. In *Findings of the Association for Computational Linguistics: NAACL 2025*.

[90] Jing Yang and Anderson Rocha. 2024. Take It Easy: Label-Adaptive Self-Rationalization for Fact Verification and Explanation Generation. In *IEEE Workshop on Information Forensics and Security (WIFS)*.

[91] Jing Yang, Didier Vega-Oliveros, Tais Seibt, and Anderson Rocha. 2021. Scalable fact-checking with human-in-the-loop. In *IEEE International Workshop on Information Forensics and Security (WIFS)*.

[92] Barry Menglong Yao, Aditya Shah, Lichao Sun, Jin-Hee Cho, and Lifu Huang. 2023. End-to-end multimodal fact-checking and explanation generation: A challenging

dataset and models. In *ACM SIGIR Conference on Research and Development in Information Retrieval*.
[93] Yejun Yoon, Jaeyoon Jung, Seunghyun Yoon, and Kunwoo Park. 2024. HerO at AVeriTeC: The Herd of Open Large Language Models for Verifying Real-World Claims. In *Fact Extraction and VERification Workshop (FEVER)*.
[94] Xinyi Zhou, Ashish Sharma, Amy X. Zhang, and Tim Althoff. 2024. Correcting misinformation on social media with a large language model. arXiv:2403.11169.
[95] Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and resolution of rumours in social media: A survey. *ACM Computing Surveys (CSUR)* (2018).

Appendix

Ethics Considerations

In our work, we carefully evaluated the ethical implications of our methodological choices on our stakeholders, who are potential users of the system, platforms whose search infrastructure we used to query and retrieve documents, individuals or groups that are the subject of misinformation claims, and the professional fact-checkers we hired as part of our user study. In hiring professional fact-checkers, we made sure to pay a competitive rate for their time ($45.79/hr). The claims we asked each fact-checker to confirm were selected to be representative of their daily workflow, and therefore no more harmful than what they might encounter in the course of their normal work. We intentionally scoped RealTiCheck to not output binary true or false judgments as it is not supposed to be interpreted as an "oracle of truth." Instead, it presents whether a claim is supported or not supported by the evidence retrievable at the time. While RealTiCheck's judgement is intended to aid fact-checkers' workflows, we stress we do *not* seek to replace critical, human-driven fact-checking work. We examine many user-configurable choices (evidence sourcing, filtering, aggregation) precisely so that deployments can adapt the system rather than defer to it. All data in our study comes from publicly accessible news websites and web search results. We did not attempt to bypass security measures or scrape private spaces in line with best practices [69]. Our system makes approximately 5 web requests to search engines per claim, which aligns with the number of requests a professional fact-checker makes when conducting their work. A potential risk is that our system may inherit biases from the sources it draws on. To alleviate this, we explicitly filter sources and analyse how source selection might shape outcomes. Since we are studying politically sensitive claims and evaluating systems that can influence public discourse, we were deliberate in designing the system to avoid potential misuse outside of its intended use-case. To this end, our system is designed to preserve the autonomy of users. Subjects of misinformation claims may themselves be harmed due to the sharing and distribution of claims in this publication. Both for clarity and reproducibility of our work, we have chosen to include the exact claims (with names of public figures) in our paper text and associated artifact, considering that such claims are already widely spread on the Internet and our intention is to share them to improve future fact-checking systems.

Open Science

We are committed to sharing our data and code with researchers seeking to conduct future work in real-time fact-checking. We release the full implementation of RealTiCheck, which includes claim detection and selection, query generation, evidence retrieval, filtering, entailment, and aggregation. We also include scripts to reproduce our microbenchmarking experiments, study instruments from our fact-checker study including the claims and survey materials and the news domains used for filtering, are all made available in our GitHub repository: https://github.com/Arshiaarya/RealTiCheck.

9 Models Comparison

We report a comparison across different LLMs tested with the AVeriTeC microbenchmark. Most LLMs perform roughly equivalently through entailment, with GPT-4o performing slightly better than the others across all configurations. As such, we use GPT-4o for our experiments.

10 Search Engine Comparison (Google vs. Bing)

To isolate the impact of the web search engine in our real-time pipeline, we re-ran our pipeline on same set from our real time experiment, holding all other components fixed. We compare Google Search vs. Bing under the *relevance-only* filter with document chunking (our default real-time setting). We report macro-averaged Precision/Recall/F1 over claims for which the system produced a decision; *N* shows the number (and fraction) of claims contributing to metrics under each configuration.

	M-P	M-R	M-F1	N
Google Search	.70	.74	.71	170 (98%)
Bing	.64	.63	.64	139 (80%)

Table 11: Entailment Results - Search Engine Choice (Real-Time, Relevance-Only, Document Chunking). With all else fixed, Google yields higher macro F1 (+.07), primarily from higher recall (+.11), and covers more claims (+18% absolute coverage).

Takeaways. (1) *Coverage:* Google returns sufficient evidence for more claims (*N*=170 vs. 139). (2) *Quality:* End-to-end performance improves notably with Google, driven largely by recall (.74 vs. .63) while maintaining higher precision (.70 vs. .64). Overall macro-F1 rises from .64 (Bing) to .71 (Google), as seen in Table 11.