PLEDGETRACKER: A System for Monitoring the Fulfilment of Pledges

Yulong Chen, Zhenyun Deng, Andreas Vlachos
University of Cambridge
{yc632, zd302, av308}@cam.ac.uk

Michael Schlichtkrull
Queen Mary University London
m.schlichtkrull@qmul.ac.uk

David Corney, Nasim Asl, Joshua Salisbury, Andrew Dudfield
Full Fact
{firstname.lastname}@fullfact.org

Abstract

Existing methods simplify the pledge monitoring task into a document classification task, overlooking its dynamic temporal and multi-document nature. To address this issue, we introduce PLEDGETRACKER, a system that formulates pledge monitoring as structured event timeline construction. PLEDGETRACKER consists of three core components: (1) a multi-step evidence retrieval module; (2) a timeline construction module and; (3) a fulfilment filtering module, enabling us to capture the evolving nature of the task. We evaluate PLEDGETRACKER in collaboration with professional fact-checkers in real-world workflows, showing its superior effectiveness over Google search and GPT-4o with web_search.

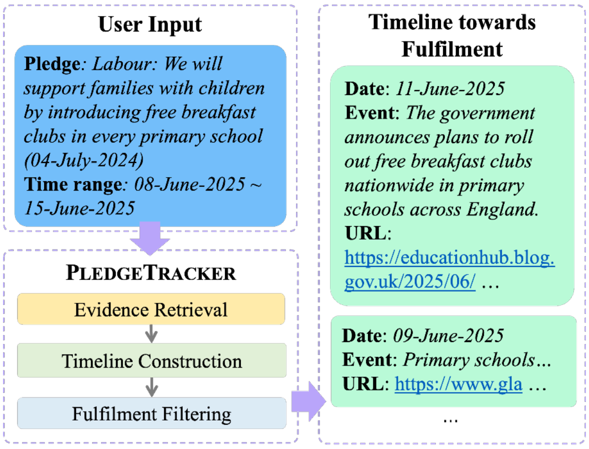

Figure 1: Overview of PLEDGETRACKER.

1 Introduction

Political pledges are commitments and governance plans made by political parties or candidates, especially during their election campaigns, which aim to promote their policies (Costello and Thomson, 2008; Dupont et al., 2019). Monitoring the fulfilment of pledges helps measure government performance, reinforcing transparency in democracy and accountability. However, this task typically requires fact-checkers to retrieve and analyse relevant documents regularly (e.g., daily or weekly) (Duval and Pétry, 2020; Fornaciari et al., 2021; Sahnan et al., 2025), which is resource-intensive, motivating the need for automated systems.

Recent work treats pledge monitoring as a *document-level* classification problem (Seki et al., 2024), by identifying whether a single article supports a pledge or not, overlooking the dynamic and long-term nature of pledge fulfilment. A political pledge is a strategic commitment, which is usually fulfilled via a sequence of actions and milestones (e.g., "*build 100 new schools in the UK by 2027*" materialises via local actions such as "*50 schools in England*" or incremental milestones like "*30 schools by 2025*"). Furthermore, the pledge status is temporal and dynamic in nature. It can evolve when new evidence emerges (e.g., the exit and re-entry into international agreements). Thus the task requires collecting and reasoning over temporally distributed evidence from multiple documents.

These requirements distinguish pledge monitoring from conventional fact-checking (Guo et al., 2022; Schlichtkrull et al., 2023; Iqbal et al., 2024). Although fact-checking also collects evidence from multi-document, it typically focuses on verifying whether a claim is supported by evidence *before* when the claim was made (Konstantinovskiy et al., 2021). Thus, the verdict is unlikely to change as those claims are about facts or knowledge that have already happened, except for corrections due to errors. In contrast, pledge monitoring aims to track how the fulfilment of a pledge evolves. Moreover, unlike the static labels of fact-checking output, pledge monitoring requires the output to reflect incremental progress over time. As such, the needs of end-users go beyond static labels, calling for structured, time-aware output.

To address these issues, we introduce PLEDGETRACKER, a retrieval augmented generation

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 839–850
November 4-9, 2025 ©2025 Association for Computational Linguistics

(RAG)-based system for monitoring the fulfilment of political pledges by extracting timelines from online documents. As shown in Figure 1, PLEDGETRACKER consists of three core components in a multi-step framework: (1) an evidence retrieval module collects and identifies relevant documents through multi-step retrieval; (2) a timeline construction module identifies and extracts key event descriptions and their timestamps from multiple relevant documents; (3) a fulfilment filtering module determines relevant events, and assembles them into a temporally-structured timeline (Hu et al., 2024). For the development of the latter, we construct an annotated dataset covering 1,559 event descriptions across 50 pledges, where each event is labelled regarding its relevance to fulfilment.

To demonstrate the effectiveness, we evaluate PLEDGETRACKER in collaboration with professional fact-checkers from Full Fact, in their real-life workflows where evidence continuously evolves. Our system achieves 0.641 F_1 in identifying fulfilment events in a real-world evaluation. Moreover, our further analysis finds PLEDGETRACKER to be more accurate in retrieving useful evidence URLs (0.78 F_1) than Google Search (0.23 F_1) and `GPT-4o` with `web_search` (0.03 F_1), both of which are part of the modules in PLEDGETRACKER. Qualitative feedback suggests that PLEDGETRACKER brings useful events to the attention of the fact-checkers that would have otherwise been missed. We publicly release PLEDGETRACKER[1] and our annotation to facilitate the task of pledge monitoring.

2 Pledge Monitoring

Drawing inspiration from fact–checking organisations like Full Fact's Government Tracker[2], pledge monitoring refers to the task of fulfilling promises with actions, i.e., when, how, and to what extent those promises are being fulfilled. We formulate this task as constructing an event timeline that reflects the progress regarding a pledge.

Formally, given a pledge $p = (p_s, p_d, p_g, p_c)$, where p_s is the pledge speaker (e.g., a political party such as *Labour*), p_d is the pledge date (i.e., when it is made), p_g is the geographic scope (e.g., *the UK*), and p_c is the pledge claim (e.g., "*We will ban trail hunting*"), and a monitoring time range $r = (r_s, r_e)$, where r_s and r_e are the start date and end date, respectively, the system $\mathcal{S}$ is asked to generate a timeline T:

$$T = \mathcal{S}(p, r), \qquad (1)$$

where T is the timeline (possibly empty if no progress has been made). For a non-empty $T = \{(e, t, url)\}$, each event description e_i is associated with a timestamp t_i and its source URL url_i, with the full set sorted in order, i.e., for all $i < j$, we have either $t_i \leq t_j$ (chronological) or $t_i \geq t_j$ (reverse chronological). Timeline T captures incremental progress and setbacks over time.

3 PLEDGETRACKER

As shown in Figure 2, PLEDGETRACKER is a RAG-based system consisting of three modules: an evidence retrieval module $\mathcal{R}$, a timeline construction module $\mathcal{T}$, and a fulfilment filtering module $\mathcal{F}$, i.e., $\mathcal{S} = \{\mathcal{R}, \mathcal{T}, \mathcal{F}\}$. Given a pledge and the time range, we first collect a set of documents using the evidence retrieval module: $D = \mathcal{R}(p, r)$. Then, based on the retrieved documents and the pledge, the timeline construction module extracts all possible events and their timestamp: $E = \mathcal{T}(D, p)$. Finally, the fulfilment filtering module selects the subset of events most useful to monitor the pledge, producing the final timeline: $T = \mathcal{F}(E, p, r)$. The subsequent subsections provide a detailed description of the corresponding modules.

3.1 Evidence retrieval

Following recent work on evidence retrieval that uses a multi-round retrieval strategy (Liao et al., 2023; Yang et al., 2024; Liu et al., 2024), PLEDGETRACKER's retrieval component is progressively expands and refines the document set D in multiple rounds of interaction and question-guided augmentation.

Given a pledge p and a target monitoring time range r, we first perform an initial web search using Google custom search API.[3] In particular, we construct a query string such as *"Labour: We will ban trail hunting (04-Jul-2024)"*, conditioned by the geographic scope p_g and the date range (r_s, r_e). As these results can often be sparse or incomplete, we further extract key noun phrases (e.g., "*trail hunting*") from the pledge content p_c using `spaCy`[4] as additional search queries. Given the retrieved

[1] https://huggingface.co/spaces/PledgeTracker/Pledge_Tracker

[2] https://fullfact.org/government-tracker/

[3] https://developers.google.com/custom-search/

[4] https://spacy.io/

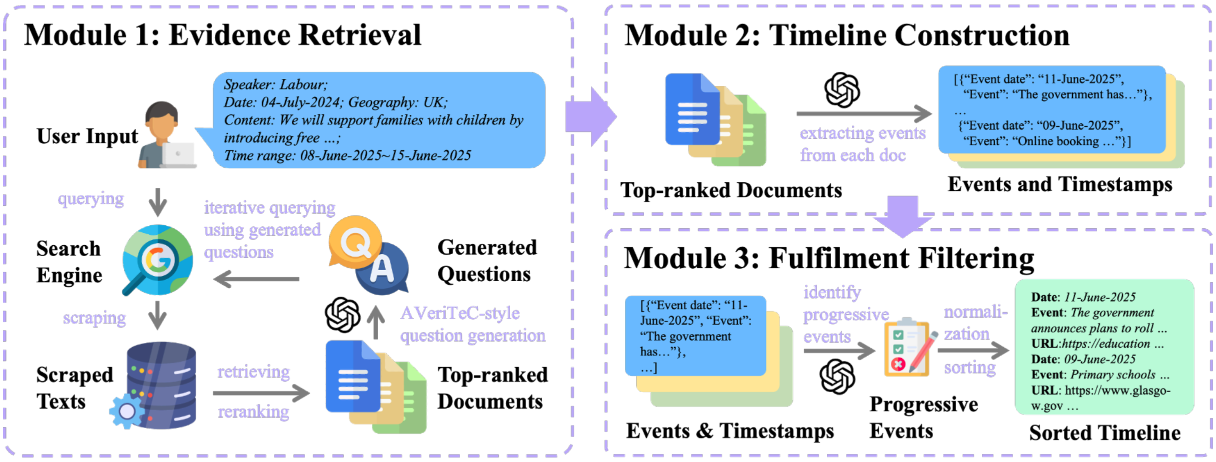

Figure 2: The architecture and workflow of PLEDGETRACKER.

URL results, we obtain the corresponding textual documents using `trafilatura` (Barbaresi, 2021), a library for web crawling and text extraction.

To guide deeper retrieval, we further incorporate question-driven augmentation based on retrieved evidence. Following Yoon et al. (2024), we first generate a set of hypothetical documents, which simulate possible evidence. We then use those hypothetical documents to retrieve sentence-level evidence from the scraped texts using `bm25`, and re-rank the evidence based on their semantic similarity computed with `SFR-Embedding-2_R`.[5] For each top-ranked evidence, we generate the corresponding clarification question that explicitly targets different aspects of the pledge (e.g., *"Is Labour planning to implement a central reporting mechanism for reporting potential animal welfare offences?"*). These questions are then used as new search queries for the next round of retrieval. Both the hypothetical document generation and question generation are performed by `Llama-3.1-8B-instruct` (Grattafiori et al., 2024) using in-context learning (ICL) examples from AVeriTeC (Schlichtkrull et al., 2023). The details can be found in Appendix A.1.

Finally, after multiple rounds of retrieval,the evidence retrieval module returns a set of top-ranked evidence. We then collect and deduplicate the document texts and corresponding URLs to construct the final D for the timeline construction module as described in the next subsection.

3.2 Timeline Construction

Rather than relying on predefined schemas (Minard et al., 2015), we adopt a generative extraction approach using `GPT-4o` (Hurst et al., 2024), which allows for more flexible identification of events (Gao et al., 2023; Chen et al., 2024a; Qorib et al., 2025).

In particular, we prompt the model using few-shot ICL examples consisting of document–event pairs that we annotated manually, and constrain the model output to follow the JSON format. Given a pledge p and each document $d_i \in D$, we construct a prompt that includes the document's metadata (e.g., title and publication date), the article body, and the pledge text, in order to generate relevant event descriptions (e.g., *"A petition is rejected because there is already a similar petition about banning trail hunting."*). Moreover, since event timestamps mentioned in the text may be expressed in various terms (e.g., publication date: `08-Jul-2024`, event temporal-related phrase: "`two days ago`"), we prompt the model through ICL to generate the corresponding absolute date (e.g., `06-Jul-2024`) if possible, or a relative date (e.g., `Last month (relative to 01-Jul-2024)`). The details can be found in Appendix A.2.

After processing all documents from D, we further sort the events by their dates. We normalise the timestamps using a rule-based parser that handles a wide range of temporal expressions (e.g., locating "`Autumn 2023`" into "`01-09-2023`"). Finally, this module returns a set of candidate events E.

3.3 Fulfilment Filtering

In practice, we find that not all events in E are informative or relevant to monitoring the fulfilment of the pledge. Although they are extracted from top-ranked documents, many events provide only contextual or background information (e.g., *What does the pledge mean?*), rather than concrete progress

[5] `https://huggingface.co/Salesforce/`

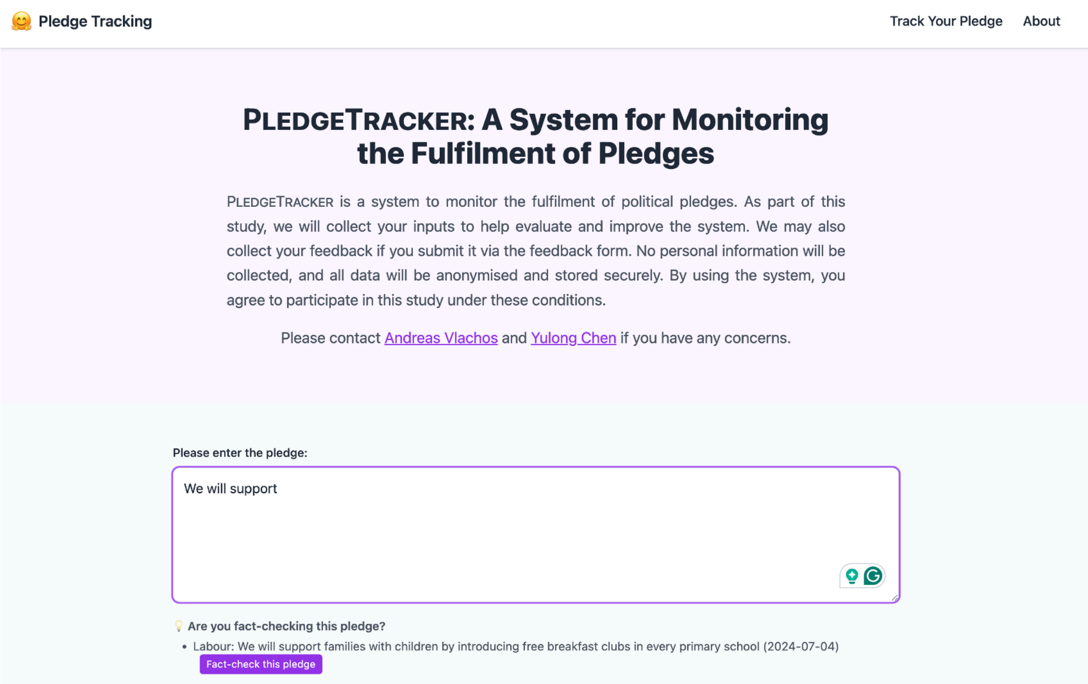

Figure 3: The user input interface of PLEDGETRACKER.

to fulfilling the pledge (*What progress has been made?*).[6] For example, while the event "*Critics claim trail hunting is being used as a 'smokescreen' for illegal fox hunting activities*" is related to "*trail hunting*", it does not contain any useful information about the actions that were taken. To address this, we developed the fulfilment filtering module $\mathcal{F}$ to filter the events to be included in the timeline.

To support the fulfilment filtering, we construct a dataset focusing on the task. We begin with a set of 50 pledges selected from FullFact government tracker, which are from the Labour Party's manifesto for the 2024 UK general election. We then use the PLEDGETRACKER (without fulfilment filtering) to retrieve all potentially related events from the time each pledge was made (starting on 4 July 2024) up to the time when the timeline was generated (March 2025). For each pledge, a professional fact-checker, who was familiar with it, examined the generated timeline and evaluated whether each event and its timestamp were useful or not, with the help of the corresponding URL. In particular, we define an event and its timestamp as *useful* in assessing fulfilment if it (1) is factually consistent with the source document, (2) contains a correctly inferred timestamp, and (3) contributes to the fulfilment of the pledge. If any of these criteria are not met, the event is labelled as *not useful*. In total, we collect 1,559 annotated instances, where each instance consists of a pledge, an event description, a timestamp, the original URL, and a binary usefulness label. In particular, our analysis shows that only 26.63% of them are useful in monitoring the fulfilment of the corresponding claims, which demonstrates the necessity of fulfilment filtering.

During testing, given each e_i from E, we ask `GPT-4o` to label each extracted event as either *useful* or *not useful* in assessing fulfilment using ICL examples from our annotation. The resulting timeline provides a clear and interpretable progression of pledge fulfilment over time. The details can be found in Appendix A.3.

4 User Interface Design

We build the PLEDGETRACKER demo system on Hugging Face Space (Nvidia A100) using Flask. Using the interface, users enter a pledge, specify the speaker, pledge date, and time range, and initiate the system by clicking the "*Let's track!*" button (Figure 3).

Once the input data is submitted, PLEDGETRACKER starts the multi-stage pipeline as detailed in §3. The system will start collecting evidence, generating the timeline, and identifying fulfilment events using ICL instances from our annotation, showing relevant status updates (Figure 4).

[6] https://fullfact.org/government-tracker/hillsborough-law-candour-duty/

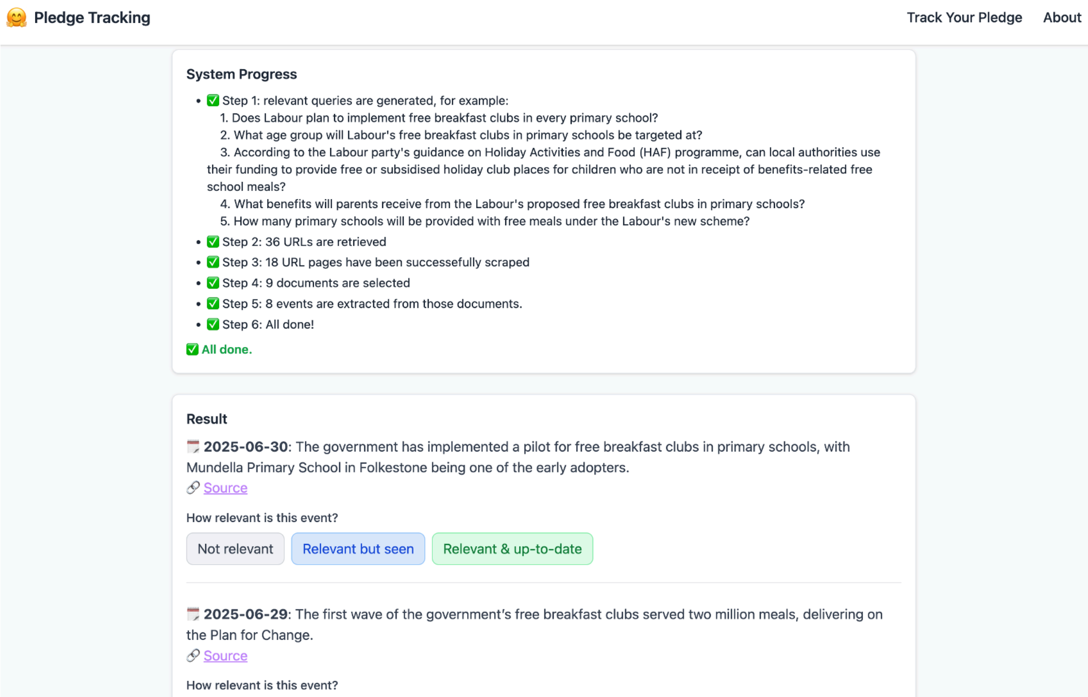

Figure 4: The output timeline interface of PLEDGETRACKER.

Finally, PLEDGETRACKER presents the timeline, where each event is associated with an event date, an event description, and the original source link. To support iterative refinement for analysis and future work, the demo system enables users to provide feedback on the usefulness of each event.

PLEDGETRACKER also supports matching pledges against previously checked ones. When the user enters a new pledge, the system automatically searches for similar pledges among the pledges already checked by the system, using `TF-IDF` and shows the top suggestions based on their similarities. For suggested pledges, the system will re-use previously retrieved results (from an initial web search) to accelerate the process and enable more accurate fulfilment filtering by selecting corresponding annotated data.

5 Experiments

We perform two kinds of quantitative evaluation: offline, using our annotated data, and in real-world use with professional fact-checkers. In particular, we first demonstrate offline the effectiveness of fulfilment filtering (§5.1), and then evaluate the full PLEDGETRACKER in real-world use (§5.2) and show its comparison with existing tools (§5.3). We further present qualitative analysis in §5.4.

	Train	**Dev**	**Test**
useful (%)	20.86	33.33	37.12
non-useful (%)	79.14	66.67	62.88
event/pledge	43.14	24.90	20.06

Table 1: Statistics for the fulfilment filtering annotation.

	P	**R**	F_1
ROBERTA	0.517	0.224	0.313
`Llama`	0.544	0.507	0.525
`GPT-4o`	0.509	0.836	0.633

Table 2: Results on fulfilment filtering.

5.1 Effectiveness of Fulfilment Filtering

As described in §3.3, we collect 1,559 instances for fulfilment filtering, which are divided into training (949), development (249) and test (361) sets based on pledges. Table 1 shows their statistics. We note the distribution difference across data splits due to fulfilment varying across pledges. We conduct experiments using three models: (1) ROBERTA-large (Liu et al., 2019) with full-parameter fine-tuning; (2) `Llama-3-8B` (Grattafiori et al., 2024) trained using instruction-based LoRA tuning (Hu et al., 2022) and; (3) `GPT-4o` with ICL prompting. Given a pledge and an associated event, each model is asked to assign a binary label indicating whether the event is useful in assessing fulfilment.

System	Pledge-level			URL-level			Novelty
	P	R	F_1	P	R	F_1	
PLEDGETRACKER	0.83	0.74	0.76	0.93	0.68	0.78	36
Google Search	0.32	0.08	0.12	0.50	0.15	0.23	5
GPT-4o with web_search	0.08	0.01	0.01	1.00	0.02	0.03	1

Table 3: Overall retrieval performance. Pledge-level: results first averaged per pledge, then averaged. URL-level: results averaged across all URLs. Novelty: the number of unique useful URLs retrieved by a system.

As shown in Table 2, GPT-4o achieves the best performance, with an F_1 score of 0.633. It suggests that, compared with RoBERTa and Llama, GPT-4o is better at capturing potential fulfilment signal. The main challenge lies in the imbalanced data distribution of the pledge monitoring data. As mentioned, the fulfilment events can be sparse in the real world, while most events lack concrete evidence of progress (c.f. Table 1).

5.2 Evaluation in Real-world Use

After deploying the full version of PLEDGETRACKER, we evaluate the system in a *real-world* setting with Full Fact fact-checkers. In particular, our evaluation was conducted from 12 June to 08 September 2025, monitoring 68 pledges from the Labour Party's 2024 UK election manifesto. Each timeline is generated over a time range of the past 7 days. As some pledges were monitored multiple times at different times in the evaluation period, we collected 113 timelines in total. Two professional fact-checkers (paper co-authors Nasim Asl and Joshua Salisbury), who were responsible for the corresponding pledges in their daily work, evaluate the usefulness of each event, using the criteria described in §3.3. We continue to present *all* candidate events, including both those retained and those filtered out, to the fact-checkers. This setup enables a direct comparison between the PLEDGETRACKER's filtering decisions and human judgments. During the evaluation, the fact-checkers select one of three labels: not_relevant, relevant_seen, and relevant_update. The label relevant_update indicates that the event is new to the fact-checkers and useful for fulfilment tracking, relevant_seen means that the event is useful and temporally appropriate, and meanwhile, fact-checkers already know about it. We therefore treat both relevant_seen and relevant_update as *useful* in our evaluation, since our goal is to assess whether the system can accurately surface relevant fulfilment evidence, regardless of whether the annotator had seen it from other sources. In total, 513 events were evaluated across 68 timelines.

Generally, PLEDGETRACKER achieves 0.764 precision, 0.553 recall and 0.641 F_1, demonstrating that it can identify fulfilment events with reasonably high performance in a real-world setting. Compared to the offline results in §5.1, the full system shows higher precision. This can be partly because the full system benefits from using the full annotation set for ICL prompting. Meanwhile, recall slightly decreases, which can be because the time range (past 7 days) is narrower, resulting in sparser fulfilment. In particular, for 513 events, we manually identify 152 fulfilment events (29.63%), which is lower than in the offline evaluation (37.12%).

5.3 Comparison with Existing Tools

We compare PLEDGETRACKER with two other tools that are often used for pledge monitoring: (1) Google Search and; (2) GPT-4o with web_search. In particular, we collect 13 pledge monitoring requests (from 12 June to 22 June 2025) from the evaluation in §5.2 that received at least one fulfilment event according to the fact-checker's judgment. We use the aforementioned two tools to return top-ranked evidence (Appendix C), and ask fact-checkers to evaluate them. Since they cannot directly return timelines, the evaluation focuses on *whether the retrieved URLs include events useful in assessing fulfilment*. For each pledge monitoring request, we first pool all URLs returned by the three systems, remove duplicates, and have professional fact-checkers label each URL. We take all *useful* URLs for a given request as the ground truth set and evaluate their performance as shown in Table 3.

Overall, PLEDGETRACKER retrieves 68% of all manually identified evidence with 0.93 precision and 0.78 F_1 at the URL level. It also contributes 36 unique, useful URLs that other systems fail to find. Compared to Google Search (0.15 recall), PLEDGETRACKER benefits from the question-driven iterative retrieval using question generation, which aligns with findings from the

ID	Pledge claim	Date	Event description, timestamp and URL
1	Labour will end the VAT exemption and business rates relief for private schools	2025-06-13	Private school families lost their High Court challenge against the Government over the VAT policy on fees. 2025-06-13. [URL]
2	Labour will capitalise Great British Energy with £8.3 billion, over the next parliament	2025-06-11	The government is delivering a new generation of publicly owned clean power. Great British Energy and Great British Energy–Nuclear will together invest more than £8.3 billion over the SR in homegrown clean power. 2025-06-11. [URL]

Table 4: Events that led to updates in Full Fact's pledge pages. The Date here refers to when the monitoring was requested. The time range is set to the past 7 days. We attach the hyperlink (URL) for reference.

AVeriTeC (Schlichtkrull et al., 2023, 2024). It is worth noting that PLEDGETRACKER has higher precision than Google Search (0.50), indicating the effectiveness of our other modules. Moreover, GPT-4o shows very poor performance in this task (0.03 F_1 at URL level). In our evaluation, we find that GPT-4o is less sensitive to temporal constraints. In particular, although GPT-4o returns 61 URLs in total, only 1 is within the correct time range.

5.4 Qualitative Feedback

The two Full Fact fact-checkers who conducted the human evaluation in §5.2 also provided some qualitative feedback. From their feedback and specific examples as shown in Table 4, we observe certain scenarios where the system has been helpful.

First, PLEDGETRACKER captures useful events that may otherwise be overlooked. In Table 4 case 1, it alerted fact-checkers to news that had not gained much coverage in the media, a High Court ruling. Although this event did not change the verdict of the pledge, it led to an update to the pledge page on the removal of the VAT exemption for private schools, as the page previously said the appeal was taking place. Second, PLEDGETRACKER assists in timely event identification. In Table 4 case 2, PLEDGETRACKER found that the investment would be split between Great British Energy and Great British Nuclear, on the same day the government's 2025 Spending Review was released. This early signal enabled them to update the pledge page promptly and contact the UK government for further clarification. Third, PLEDGETRACKER helps surface legislative and political signals that inform future developments. Fact-checkers found PLEDGETRACKER could highlight the names of bills and draft legislations associated with pledges, and trace their mentions across time in official communications. For example, it surfaces passing remarks by politicians, indicating when legislation or announcements could be expected, which was not previously captured through routine monitoring. Overall, they reported that PLEDGETRACKER greatly contributes to their workflow.

In addition to these strengths, fact-checkers also noted occasional hallucinations in the event descriptions, for example, the generated events can be inconsistent with the source documents. To mitigate this known limitation of LLMs (Zhang et al., 2023; Chen et al., 2024b), PLEDGETRACKER is designed to explicitly include source URLs for each event, allowing fact-checkers to verify the underlying evidence when necessary.

6 Conclusion and Future Work

We presented PLEDGETRACKER, the first end-to-end system that formulates pledge monitoring as the construction of temporally ordered timelines. By iteratively collecting evidence from online, with generative timeline construction and fulfilment filtering, PLEDGETRACKER captures incremental evidence and generates more interpretable outputs. We integrated the system into professional fact-checkers' real-life workflows, and found PLEDGETRACKER achieved an F_1 of 0.641 in identifying fulfilment events. Our further comparison with Google Search and GPT-4o with web_search, demonstrating the superior performance of PLEDGETRACKER for pledge monitoring.

Limitations

The limitations of PLEDGETRACKER can be stated from four perspectives. First, PLEDGETRACKER is built on the basis that pledges have already been identified and normalised, and therefore it does not address the task of automatically extracting and decontextualising pledges from manifestos (Deng et al., 2024; Panchendrarajan and Zubiaga, 2024). Second, our evaluation focuses on pledges from UK political parties. However, its effectiveness in other linguistic or institutional contexts remains to be further explored (Zhang et al., 2024; Turk et al., 2025). Third, our evidence retrieval relies heavily on the Google Custom Search API, limiting its evidence coverage with potential ranking bias, and quota constraints. Fourth, due to the limited resources, we could not perform large-scale training and thus use small models and LLM APIs for implementing PLEDGETRACKER.

Ethical Considerations

Our work involves human annotation and evaluation as stated in §3.3, §5.2, and §5.3. These two annotators are professional fact-checkers and the co-authors of this paper. Their background information is provided in Appendix B.

We acknowledge that LLMs exhibit political biases (Chalkidis and Brandl, 2024); however, we mitigate these by using RAG (Lewis et al., 2020; Ram et al., 2023) and providing the URLs of the sources used for the timeline construction, so that users can verify the output themselves. Furthermore, we evaluated the system with fact-checkers from Full Fact, which is a signatory to the International Fact-Checkers Network code of principles (`https://ifcncodeofprinciples.poynter.org/the-commitments`) that stipulates that they need to be impartial in their work.

The release of our demo has been approved by the Ethics Review Committee[7] at the Department of Computer Science and Technology, University of Cambridge, under the CC-BY-NC license.

Acknowledgements

We appreciate the program chairs, area chair, and reviewers from EMNLP 2025 System Demonstration for their insightful feedback. We would like to thank Chenxi Whitehouse and Tom Stafford for their contributions to an earlier version of this project. This work is supported by the ERC grant AVERITEC (GA 865958). Full Fact's work is funded by the JournalismAI Innovation Challenge, supported by the Google News Initiative. Michael is further supported by the Engineering and Physical Sciences Research Council (grant number EP/Y009800/1), through funding from Responsible AI UK (KP0016).

[7] `https://www.cst.cam.ac.uk/local/policy/ethics`

References

Adrien Barbaresi. 2021. Trafilatura: A web scraping library and command-line tool for text discovery and extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 122–131, Online. Association for Computational Linguistics.

Ilias Chalkidis and Stephanie Brandl. 2024. Llama meets EU: Investigating the European political spectrum through the lens of LLMs. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 481–498, Mexico City, Mexico. Association for Computational Linguistics.

Ruirui Chen, Chengwei Qin, Weifeng Jiang, and Dongkyu Choi. 2024a. Is a large language model a good annotator for event extraction? In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 17772–17780. AAAI Press.

Yulong Chen, Yang Liu, Jianhao Yan, Xuefeng Bai, Ming Zhong, Yinghao Yang, Ziyi Yang, Chenguang Zhu, and Yue Zhang. 2024b. See what LLMs cannot answer: A self-challenge framework for uncovering LLM weaknesses. In *First Conference on Language Modeling*.

Rory Costello and Robert Thomson. 2008. Election pledges and their enactment in coalition governments: A comparative analysis of ireland. *Journal of Elections, Public Opinion and Parties*, 18(3):239–256.

Zhenyun Deng, Michael Schlichtkrull, and Andreas Vlachos. 2024. Document-level claim extraction and decontextualisation for fact-checking. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11943–11954, Bangkok, Thailand. Association for Computational Linguistics.

Julia C Dupont, Evelyn Bytzek, Melanie C Steffens, and Frank M Schneider. 2019. Which kind of political campaign messages do people perceive as election pledges? *Electoral Studies*, 57:121–130.

Dominic Duval and François Pétry. 2020. Citizens' evaluations of campaign pledge fulfillment in canada. *Party Politics*, 26(4):437–447.

Tommaso Fornaciari, Dirk Hovy, Elin Naurin, Julia Runeson, Robert Thomson, and Pankaj Adhikari. 2021. "we will reduce taxes" - identifying election pledges with language models. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3406–3419, Online. Association for Computational Linguistics.

Jun Gao, Huan Zhao, Changlong Yu, and Ruifeng Xu. 2023. Exploring the feasibility of chatgpt for event extraction. *ArXiv preprint*, abs/2303.03836.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *ArXiv preprint*, abs/2407.21783.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Qisheng Hu, Geonsik Moon, and Hwee Tou Ng. 2024. From moments to milestones: Incremental timeline summarization leveraging large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7232–7246.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *ArXiv preprint*, abs/2410.21276.

Hasan Iqbal, Yuxia Wang, Minghan Wang, Georgi Nenkov Georgiev, Jiahui Geng, Iryna Gurevych, and Preslav Nakov. 2024. OpenFactCheck: A unified framework for factuality evaluation of LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 219–229, Miami, Florida, USA. Association for Computational Linguistics.

Lev Konstantinovskiy, Oliver Price, Mevan Babakar, and Arkaitz Zubiaga. 2021. Toward automated factchecking: Developing an annotation schema and benchmark for consistent automated claim detection. *Digital Threats*, 2(2).

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Hao Liao, Jiahao Peng, Zhanyi Huang, Wei Zhang, Guanghua Li, Kai Shu, and Xing Xie. 2023. Muser: A multi-step evidence retrieval enhancement framework for fake news detection. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4461–4472.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692.

Zheng Liu, Yujia Zhou, Yutao Zhu, Jianxun Lian, Chaozhuo Li, Zhicheng Dou, Defu Lian, and Jian-Yun Nie. 2024. Information retrieval meets large language models. In *Companion Proceedings of the ACM Web Conference 2024*, pages 1586–1589.

Anne-Lyse Minard, Manuela Speranza, Eneko Agirre, Itziar Aldabe, Marieke van Erp, Bernardo Magnini, German Rigau, and Rubén Urizar. 2015. SemEval-2015 task 4: TimeLine: Cross-document event ordering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 778–786, Denver, Colorado. Association for Computational Linguistics.

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*, 7:100066.

Muhammad Reza Qorib, Qisheng Hu, and Hwee Tou Ng. 2025. Just what you desire: Constrained timeline summarization with self-reflection for enhanced relevance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25065–25073.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Dhruv Sahnan, David Corney, Irene Larraz, Giovanni Zagni, Ruben Miguez, Zhuohan Xie, Iryna Gurevych, Elizabeth Churchill, Tanmoy Chakraborty, and Preslav Nakov. 2025. Can llms automate fact-checking article writing?

Michael Schlichtkrull, Yulong Chen, Chenxi Whitehouse, Zhenyun Deng, Mubashara Akhtar, Rami Aly, Zhijiang Guo, Christos Christodoulopoulos, Oana Cocarascu, Arpit Mittal, and 1 others. 2024. The automated verification of textual claims (averitec) shared task. In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*, pages 1–26.

Michael Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. 2023. Averitec: A dataset for real-world claim verification with evidence from the web. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Yohei Seki, Hakusen Shu, Anaïs Lhuissier, Hanwool Lee, Juyeon Kang, Min-Yuh Day, and Chung-Chi Chen. 2024. Ml-promise: A multilingual dataset for corporate promise verification. *ArXiv preprint*, abs/2411.04473.

Nawar Turk, Eeham Khan, and Leila Kosseim. 2025. Clac at semeval-2025 task 6: A multi-architecture approach for corporate environmental promise verification. *arXiv preprint arXiv:2505.23538*.

Diji Yang, Jinmeng Rao, Kezhen Chen, Xiaoyuan Guo, Yawen Zhang, Jie Yang, and Yi Zhang. 2024. IM-RAG: multi-round retrieval-augmented generation through learning inner monologues. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, pages 730–740. ACM.

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 *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*, pages 130–136, Miami, Florida, USA. Association for Computational Linguistics.

Caiqi Zhang, Zhijiang Guo, and Andreas Vlachos. 2024. Do we need language-specific fact-checking models? the case of chinese. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1899–1914.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, and 1 others. 2023. Siren's song in the ai ocean: a survey on hallucination in large language models. *ArXiv preprint*, abs/2309.01219.

A Implementation Details

A.1 Evidence Retrieval

Following Yoon et al. (2024), we index the training data from AVeriTeC (Schlichtkrull et al., 2023) and retrieve the top-10 most similar question-evidence pairs to the input pledge from the training corpus using BM25. These top-10 question-evidence pairs are then used as the ICL examples. In particular, the prompt is as follow:

> Your task is to generate a question based on the given claim and evidence. The question should clarify the relationship between the evidence and the claim.
>
> `{ICL_examples}`
>
> Now, generate a question that links the following claim and evidence:
>
> Claim: `{pledge_claim}`
> Evidence: `{sentence_evidence}`

We use `Meta-Llama-3.1-8B-Instruct` with a temperature of 0.6 and top-p of 0.9. We generate one question per evidence sentence.

For Google Custom Search, we set the geographic scope to the UK due to our focus on the UK election pledges. In practice, we set the iterative evidence retrieval to two rounds, to balance a good result in practice and our budgets.

A.2 Timeline Construction

We use the below prompt for event description generation and timestamp identification:

> Please only summarize events that are useful for verifying the pledge, and their dates in the JSON format.
>
> `{ICL_examples}`
>
> Please only summarize events that are useful for verifying the pledge: `{pledge}`, and their dates in the JSON format.
>
> Input:
>
> Title: `{document_title}`
> Date: `{document_date}`
> Article: `{document_text}`
>
> Output:

Please note that we use `GPT-4o` for experiments, and constrain the output (including the outputs of the ICL pairs) in the JSON format, for example:

```
{
"events": [
{
"event": "Home Secretary Yvette Cooper announces new measures to boost Britain's border security, including the recruitment of up to 100 new specialist intelligence and investigation officers at the National Crime Agency (NCA).",
"date": "2024-08-21"
},
{
"event":
"Announcement of a major surge in immigration enforcement and returns activity to achieve the highest rate of removals of those with no right to be in the UK since 2018.",
"date": "2024-08-21"
},
{
...
}]
}
```

We use 2 ICL examples to balance the length constraint and model efficiency. We set the top-p and temperature to 0.

A.3 Relevant Event Identification

We use the below prompt for identifying relevant events:

> You are given a pledge, the pledge speaker, and the date of when the pledge is made, and a key event summarized from an online article along with the date of when the event happens. Your task is to determine whether this event summary is useful to track the fulfilment of this pledge.
>
> Yes: The summary presents developments or actions that demonstrate progress (or lack thereof) towards fulfilling the pledge. It helps evaluate whether the pledge is on track or not.
>
> No: The summary only provides background or contextual information, but no progress information for evaluating the fulfilment of the pledge; Or the summary is less than or not related to the pledge.
>
> Below are examples:
>
> `{ICL_examples}`
>
> Now, please assign a label to the below instance.
>
> Input:
>
> Pledge: `{pledge}`
> Event summary: `{event}`. (Event Date: `{event_date}`)
>
> Output:

The model is expected to return `Yes` or `No`, and we also log the log-probability of the first predicted token to support confidence-based ranking.

We use at most 50 ICL examples. In particular, in our demo system, if we are checking a suggested pledge, we use their corresponding annotated data; otherwise, we randomly select instances from all annotated data. We set the top-p and temperature to 0.

B Fact-checkers' Background

Both of the fact-checkers involved in this study (Nasim and Josh) are native English speakers, educated to postgraduate level. One has worked as a fact-checker for two years and overall as a trained journalist for seven years, while the other has worked as a journalist for eight years and in fact-checking for several months.

C Setup of Google Search and GPT-4o for Real-world Evaluation

We use `GPT-4o` with the tool of `web_search`. We set the location as the UK (`GB` in `GPT-4o`), and the `search_context_size` as `high`. We use the same request for initial searching as the input, and use the below prompt to inform the model of the time range:

> Please find the recent online articles (from `{time_start}` to `{time_end}`) that can help monitor the fulfilment of the pledge. List only the article URLs, ordered by their usefulness and relevance (most useful and relevant first), one per line.
>
> `{pledge}`

Similarly, we use the same API for PLEDGETRACKER to call Google Search using the same parameters, and collect the top-10 retrieved results based on their prominence.

To ensure the retrieved URLs are within the correct time range, we further filter all URLs by examining their metadata, and use the useful URLs for evaluation.