Graph-of-Thoughts for Fact-Checking with Large Language Models

Sascha Rolinger[1], **Jin Liu**[1,2]

[1]Karlsruhe Institute of Technology, Karlsruhe, Germany
[2]FZI Research Center for Information Technology, Karlsruhe, Germany
Correspondence: sascha.rolinger@student.kit.edu, jin.liu@fzi.de

Abstract

We present a fact-checking system developed for the 2025 Automated Verification of Textual Claims (AVeriTeC) shared task, leveraging the Graph-of-Thoughts (GoT) prompting scheme. The GoT approach facilitates iterative refinement during fact-checking by conditioning question generation on previous answers and enabling the incorporation of multiple evidence documents per question, thereby mitigating the impact of factually incorrect evidence. The efficiency requirements of the shared task are addressed by restricting the width and depth of the thought graph. Additionally, an efficient stopping criterion is derived from the dataset's Not Enough Information (NEI) label. Our system utilizes fine-tuned open-source Large Language Models (LLMs) for question generation, question answering, and final verdict prediction. Empirical results demonstrate competitive performance against top-performing systems in the AVeriTeC shared task and improvements over the baseline method. Our code is publicly available[1].

1 Introduction

Automated fact-checking can be conceptualized as a textual entailment task, aiming to assess a claim's veracity based on retrieved evidence (Vlachos and Riedel, 2014). Existing automated fact-checking systems predominantly adopt a pipeline architecture, sequentially executing distinct components for claim detection, evidence retrieval, verdict prediction, and justification production (Guo et al., 2022).

The AVeriTeC dataset, introduced by Schlichtkrull et al. (2023), supports automated fact-checking with real-world claims from fact-checking articles. These claims are annotated with question-answer (QA) pairs designed to mirror a fact-checker's reasoning process. The dataset includes a Knowledge Store that identifies one or more "gold" documents for each claim, signifying their use in the original fact-check. The current AVeriTec shared task imposes two key constraints: participants must use open-source models, and fact-checking must be performed under one minute per claim. This paper describes our system for the AVeriTeC shared task based on the Graph-of-Thoughts framework.

2 Related Work

Iterative question generation for fact-checking has been explored on similar datasets (Pan et al., 2023; Wang and Shu, 2023; Zhang and Gao, 2023). Malon (2024)'s framework, developed for the AVeriTeC 2024 shared task, employs an iterative reasoning strategy that continues generating follow-up questions until the system determines sufficient evidence has been gathered. While this iterative refinement is similar to our approach, key differences exist. Malon (2024)'s system terminates when its underlying LLM determines that an adequate number of questions have been answered. In contrast, our method utilizes the NEI label, iterating until a label other than NEI is predicted or a maximum number of question rounds has been reached. Furthermore, our GoT approach explores multiple verification paths, enabling more robust handling of misleading or insufficient evidence.

While top-performing systems in the previous AVeriTeC shared task often relied on proprietary models (Rothermel et al., 2024; Ullrich et al., 2024; Park et al., 2024; Malon, 2024), with Yoon et al. (2024) being an exception, the current iteration of the shared task requires the use of open-source models for all participants.

[1]https://gitlab.kit.edu/utjwp/Fact-Checking_GoT_RAG_LLM

Proceedings of the Eighth Fact Extraction and VERification Workshop (FEVER), pages 266–273

3 Methodology

Our framework is designed to perform iterative fact-checking by building a dynamic reasoning structure. This section describes our framework, core components, their training, and the implementation details.

3.1 Graph-of-Thoughts (GoT)

Various prompting strategies for LLMs have been proposed. Among these, Besta et al. (2024) introduced GoT, a method that, similar to Chain-of-Thought (CoT) (Wei et al., 2023) or Tree-of-Thoughts (ToT) (Yao et al., 2023), models LLM thoughts as vertices and their dependencies as edges. An edge $(t1, t2)$ signifies that $t1$ serves as input for generating $t2$. Unlike other prompting methods, GoT uniquely permits the construction of an arbitrary directed graph, enabling the aggregation of individual thoughts and even entire thought chains.

The GoT framework involves two distinct graphs. The GoT itself is the central data structure that records execution results, as illustrated in Figure 2. The Graph-of-Operations (GoO) defines the algorithm responsible for generating and managing the GoT structure. Figure 1 illustrates the GoO of our system using UML-like notation, where nodes represent operations that either prompt an LLM to generate new thoughts or evaluate these thoughts through scoring, verification, or ranking.

The algorithm begins with an initial question generation phase, followed by pruning of similar questions. Subsequently, evidence is retrieved, and answers are generated. These answers can then optionally undergo a verification step. Given that multiple answers might arise from different pieces of evidence for the same underlying query, the algorithm merges these answers before forming an intermediate verdict. If the verdict is NEI and the maximum number of questioning rounds has not been reached, the algorithm iterates back to question generation. If the maximum depth is reached or the verdicts are conclusive, different reasoning branches are merged to produce a final verdict for the original claim. Figure 2 shows an annotated theoretical GoT that illustrates these algorithmic stages within the data structure.

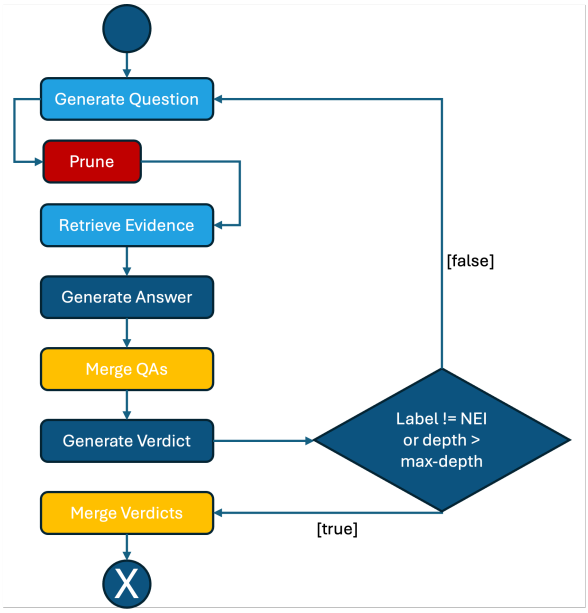

Figure 1: Control flow of the algorithm (Graph-of-Operations) used for generating and managing the Graph-of-Thoughts (GoT). Nodes indicate operations producing LLM-generated thoughts or their assessment through scoring, verification, and ranking. Color coding indicates branching operations (light blue), branch termination (red), and merging of branches (yellow)

3.2 Core Components and Training Data Preparation

Our framework consists of four core components: Question Generation, Evidence Retrieval, Question Answering, and Verdict Prediction. To achieve better performance, we fine-tuned a base LLM using the AVeriTeC dataset. Specifically, we employed Low-Rank Adaptation (LoRA) (Hu et al., 2022) to train three distinct adapters for the Question Generation, Question Answering, and Verdict Prediction tasks. The AVeriTeC dataset, in its original form, is not explicitly structured for the iterative process. Therefore, a transformation is necessary to align the dataset with the structural requirements for fine-tuning models dedicated to these iterative tasks. For sections regarding these three components, we focus on the methodology for constructing appropriate training data.

3.2.1 Question Generation

A training instance is created for each question in the dataset. The first question in the sequence requires a distinct prompt, as no prior QA pairs exist. The prompt includes the claim and its metadata for generating relevant fact-checking questions. For each subsequent question $i + 1$, the prompt is ex-

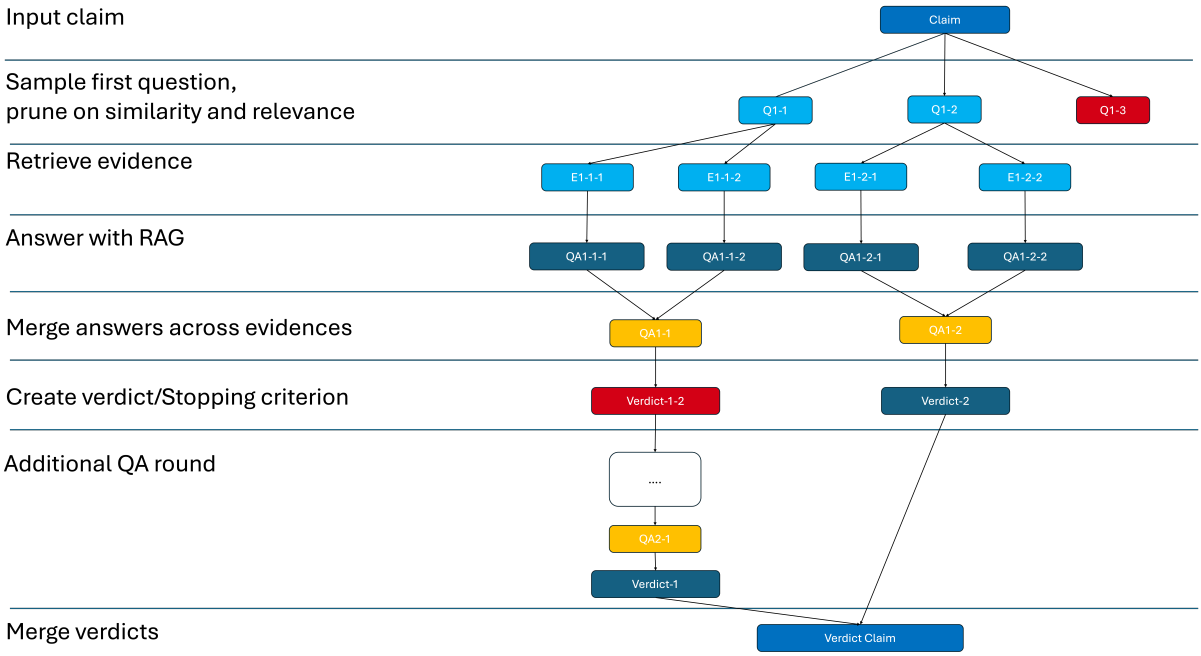

Figure 2: Illustrative example of the Graph-of-Thoughts (GoT) framework depicting the sequence of operations and their dependencies within the fact-checking process. Multiple questions are generated starting from the original claim, and each question starts a branch. Not all generated questions are explored to limit the size of the graph. For each question, multiple pieces of evidence are retrieved, and answers are generated individually. Answers can then be verified and merged. If the verdict is NEI, an additional question round is started. If all branches have a final verdict, the verdict for the claim is calculated using majority voting.

tended to include the preceding i QA pairs as additional context. This setup enables the model to condition question generation on the information gathered during earlier fact-checking steps. The target in each case is the following question in the sequence for the given claim. We show training examples for question generation in A.1.1.

3.2.2 Evidence Retrieval

Existing retrieval modules often employ multi-stage pipelines encompassing pre-processing, coarse retrieval, and re-ranking components (Zhao et al., 2022). A common retrieval pipeline involves segmenting documents into smaller chunks and encoding these with a bi-encoder to construct an efficient index. Subsequently, a cross-encoder can refine the initial retrieval results by re-ranking the top candidates. While we initially explored this standard design, its application to the AVeriTeC Knowledge Store proved computationally expansive for the shared task's constraints. Specifically, the AVeriTeC Knowledge Store averages 86,154 text chunks (each approximately 1000 characters with a 200-character overlap), making a full bi-encoder/cross-encoder pipeline excessively resource-intensive.

To allocate a greater portion of the limited time budget to the LLM generations central to our Graph-of-Thoughts (GoT) approach, we adopted a more efficient retrieval strategy. The Knowledge Store is first chunked. Then, for each claim, only the top 3000 chunks scored by BM25 relevance to the claim are retained. This pre-selection method, also employed by Ullrich et al. (2024) to reduce the size of the Knowledge Store, forms the basis for building an FAISS index (Johnson et al., 2019). This condensed index allows for more efficient querying and offers opportunities for parallelization, further optimizing the now more GPU-intensive workload. Such an efficient index structure is particularly beneficial for the GoT methodology, as the Knowledge Store is queried multiple times with questions that evolve based on the outcomes of previous queries.

3.2.3 Question Answering

Preparing the fine-tuning dataset for Question Answering introduces specific challenges due to the nature of the AVeriTeC dataset. The dataset does not provide explicit span-level annotations for pinpointing the exact evidence within the gold documents. Instead, for each question, only the iden-

tifier of the relevant gold document is supplied. To address this, the identified gold document is first segmented into manageable chunks. Subsequently, a cross-encoder-based retrieval model is employed to identify the chunk most relevant to the gold question-answer (QA) pair. Since the ground truth answer is available during training, both the question and the answer are independently used as queries to the retriever. The chunk achieving the highest relevance score from either of these queries is then selected as the candidate evidence.

Additional filtering steps are applied to mitigate noise potentially introduced by annotation errors or retrieval inaccuracies. First, a minimum relevance score threshold is applied, and any candidate evidence chunks falling below this threshold are discarded. Following this, an LLM is prompted to assess whether the ground truth answer is entailed by the retrieved evidence chunk. Each resulting training instance for the Question Answering model consists of an input concatenating the claim, metadata, the selected evidence chunk, and the question. The target output comprises the answer itself, its type (e.g., extractive, Boolean, or abstractive), and the source (metadata of evidence). We illustrate one training example for Question Answering in A.1.2.

3.2.4 Verdict Prediction

Each training instance for verdict prediction incorporates the claim, its associated metadata, and all corresponding QA pairs as input. Specific instructions are appended to this input to prompt the model to generate a justification and a veracity label, formatted in JSON. Following Liu et al. (2024), the model is guided to generate the justification before the label, enabling it to condition the final veracity assessment on the generated explanatory text. One training example is shown in A.1.3

Including additional training examples explicitly labeled as NEI is critical. These NEI instances are designed to reduce the likelihood of the model prematurely assigning a verdict of Supported, Refuted, or Conflicting Evidence/Cherry-picking when the available information is indeed insufficient. To generate these NEI examples, for each claim with n QA pairs in the AVeriTeC dataset, additional training instances are created using only the first i QA pairs, where $i \in \{1, \ldots, n-1\}$ and assigned the NEI label.

3.3 Auxiliary Components

An overview of additional components integrated into the system is provided in Table 1. The implementation of these auxiliary components is generally straightforward, primarily leveraging the models deployed for retrieval and the base LLM.

Module	Method
Question Deduplication	Bi-Encoder Filtering
Answer Merging	Prompting Base LLM
Verdict Merging	Majority Voting

Table 1: Overview of auxiliary modules and corresponding methods.

3.4 Implementation Details

The model selection is primarily constrained by the available hardware and time budget of the evaluation system, which is a g5.2xlarge EC2 instance on AWS with 23GB of GPU memory. Given that the base LLM is the central component of our approach, efficient utilization of this GPU memory was paramount. Consequently, we selected Qwen2.5-14B-Instruct-AWQ (Team, 2024; Yang et al., 2024), deploying using the vLLM (Kwon et al., 2023) inference engine. This model, due to Activation-aware Weight Quantization (AWQ) (Lin et al., 2024), fits within the available GPU memory while also reserving sufficient space for Key-Value (KV) caching, which is crucial for generation performance. The LoRA configuration employs a rank of 16, an alpha value of 32, a dropout rate of 0.05, and omits bias terms. LoRA modules are injected into all projection layers of the transformer blocks. This memory-efficient LLM deployment also leaves adequate room to deploy the bi-encoder model for various similarity calculations, including evidence retrieval, question deduplication. We chose dunzhang/stella_en_400M_v5[2] as our bi-encoder model.

4 Evaluation

We report the results for our submitted system and the average processing time per claim. The development split was processed on an Nvidia A100 with 40GB of memory. For this split, the Ev2R score (Akhtar et al., 2024) was calculated using recommended Llama-3.3-70B-Instruct (AI@Meta, 2024). Results for the test split are derived from the

[2] https://huggingface.co/Marqo/dunzhang-stella_en_400M_v5

official published leaderboard[3]. Notably, scores for the development split are significantly higher than those on the test split. This discrepancy aligns with observations from the baseline model provided by the shared task organizers, which achieved AVeriTeC scores of 0.296 on the development split and 0.2023 on the test split, respectively.

Our submitted system demonstrated notable efficiency, utilizing only a third of the maximum one-minute processing time allowed per claim. An analysis of the F1 scores reveals a disparity between supported and refuted claims, with the former achieving a comparatively lower score. This suggests that while the system is adept at refuting claims when no supporting evidence is immediately found (potentially in the initial questioning round), it may not consistently recognize when sufficient evidence has been gathered to confirm a "Supported" verdict, leading to premature termination of the evidence collection process for such claims. Our system demonstrates limited performance when classifying instances as "Not Enough Evidence" (NEI) or "Conflicting Evidence". The low prediction rate for NEI is partially a consequence of the stopping criterion in our GoT mechanism, which uses this label to trigger further question generation rather than as a final verdict. This design results in NEI being infrequently predicted, with only 10 out of 500 instances in the development set classified as such.

	Dev	Test
Ev2R recall		
Question-only	0.392	0.362
Question-answer	0.530	0.400
AVeriTeC Score	0.366	0.244
Veracity F1 Scores		
Supported	0.615	-
Refuted	0.824	-
Not Enough Evidence	0.000	-
Conflicting Evidence	0.050	-
Time Measurement		
Seconds per claim	15.4	18.5

Table 2: Comparison of Ev2R recall scores (question-only, question-answer, AVeriTeC), veracity F1 scores, and time measurements for development and test split of the 2025 AVeriTeC shared task.

5 Conclusion

We have implemented an iterative Graph-of-Thoughts (GoT) framework designed to advance fact-checking methodologies by enabling more profound and extensive exploration of evidence. Our approach has yielded competitive results, demonstrating that the incorporation of more comprehensive evidence can significantly improve fact-checking performance, even when constrained to smaller-scale, open-source LLMs. Despite these achievements, establishing a consistently reliable criterion for determining evidence sufficiency proved challenging. Consequently, fact-checking processes were either prematurely terminated or necessitated continuation for a fixed, and potentially inefficient, number of question-answering rounds.

The computational costs associated with LLM-based fact-checking, particularly within highly iterative frameworks like GoT, remain a significant hurdle. This challenge is exacerbated for operational steps where dedicated training data is scarce, requiring LLMs to be prompted with lengthy instructions, in-context examples, and intermediate reasoning outputs. The computational burden becomes especially acute near the evidence retrieval stage, where the GoT typically expands to its maximum width, demanding substantial processing resources.

Future work could focus on extending the system with several promising modules and LLM invocations. Further modules can be developed for assessing the utility of an evidence document for the fact-checking task and for verifying the entailment of a generated answer within the provided evidence. Other potential enhancements to our approach include employing an LLM for dynamic ranking of all generated thoughts, enabling more adaptive exploration of the GoT, or developing mechanisms to recover from high uncertainty in the majority voting process used for branch label determination.

Limitations

Several limitations inherent in the conducted research and the implemented system should be considered when interpreting the results and conclusions presented in this paper.

Decoding Strategy A sampling-based decoding strategy was generally employed for LLM inference. While this approach is necessary for certain components to generate diverse outputs, it introduces variability that may reduce comparability between system configurations. Greedy decoding could have provided more stable outputs across runs, potentially enabling clearer distinctions in system performance.

[3] https://fever.ai/task.html

Mismatch Between Dataset and Approach The AVeriTeC dataset is not optimally suited to the iterative, multi-hop approach investigated in this paper. Annotations in the dataset reflect reasoning chains from successful human fact-checks, offering limited opportunities for the system to learn from fact-checking failures or incomplete reasoning. The quality of the derived fine-tuning datasets is also uncertain, primarily due to the lack of gold span annotations and the presence of broken or incomplete documents in the Knowledge Store. Moreover, the Knowledge Store lacks explicit positive and negative evidence annotations, making it difficult to conduct robust retrieval experiments or to evaluate evidence selection systematically.

Comprehensiveness of Fact-Checking A notable limitation is the tendency of the dataset to contain initial gold questions that are simple rephrasings of the original claim—a concern also raised by Malon (2024). Despite annotation guidelines discouraging trivial reformulations (Schlichtkrull et al., 2023), such questions are common. Consequently, the system often follows this pattern, producing shallow question-answer sequences that may lead to shorter and less comprehensive fact-checking.

References

AI@Meta. 2024. Llama 3 model card.

Mubashara Akhtar, Michael Schlichtkrull, and Andreas Vlachos. 2024. Ev2r: Evaluating evidence retrieval in automated fact-checking. *Preprint*, arXiv:2411.05375.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. *Preprint*, arXiv:2308.09687.

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 *International Conference on Learning Representations*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for llm compression and acceleration. In *MLSys*.

Jin Liu, Steffen Thoma, and Achim Rettinger. 2024. FZI-WIM at AVeriTeC shared task: Real-world fact-checking with question answering. In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*, pages 77–85, Miami, Florida, USA. Association for Computational Linguistics.

Christopher Malon. 2024. Multi-hop Evidence Pursuit Meets the Web: Team Papelo at FEVER 2024. In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*, pages 27–36, Miami, Florida, USA. Association for Computational Linguistics.

Liangming Pan, Xinyuan Lu, Min-Yen Kan, and Preslav Nakov. 2023. QACHECK: A Demonstration System for Question-Guided Multi-Hop Fact-Checking. https://arxiv.org/abs/2310.07609v1.

Heesoo Park, Dongjun Lee, Jaehyuk Kim, ChoongWon Park, and Changhwa Park. 2024. Dunamu-ml's Submissions on AVERITEC Shared Task. In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*, pages 71–76, Miami, Florida, USA. Association for Computational Linguistics.

Mark Rothermel, Tobias Braun, Marcus Rohrbach, and Anna Rohrbach. 2024. InFact: A Strong Baseline for Automated Fact-Checking. In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*, pages 108–112, Miami, Florida, USA. Association for Computational Linguistics.

Michael Sejr Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. 2023. Averitec: A dataset for real-world claim verification with evidence from the web. In *Thirty-thh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Herbert Ullrich, Tomáš Mlynář, and Jan Drchal. 2024. AIC CTU system at AVeriTeC: Re-framing automated fact-checking as a simple RAG task. *Preprint*, arXiv:2410.11446.

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

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*, pages 6288–6304, Singapore. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Preprint*, arXiv:2201.11903.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 40 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *Preprint*, arXiv:2305.10601.

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. *Preprint*, arXiv:2410.12377.

Xuan Zhang and Wei Gao. 2023. Towards LLM-based Fact Verification on News Claims with a Hierarchical Step-by-Step Prompting Method. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 996–1011, Nusa Dua, Bali. Association for Computational Linguistics.

Wayne Xin Zhao, Jing Liu, Ruiyang Ren, and Ji-Rong Wen. 2022. Dense Text Retrieval based on Pretrained Language Models: A Survey. *Preprint*, arXiv:2211.14876.

A Appendix

A.1 Prompts

A.1.1 Training Examples for Question Generation

Figure 3 shows the training example for generating the first question. The example for subsequent question generation is illustrated in Figure 4.

A.1.2 Training Example for Question Answering

Figure 5 shows a training example for Question Answering.

Training Example for First Question Generation

Input:

```
Claim: Donald Trump delivered the largest tax cuts in
American history.
Metadata: {'claim_date': '25-8-2020', 'speaker':
'Eric Trump', 'original_claim_url': None,
'reporting_source': 'Speech at The Republican
National Convention', 'location_ISO_code': 'US'}
Context:
None
Predict the first question:
```

Target:

```
Did the 2017 tax bill deliver the largest tax cuts in
American history?
```

Figure 3: Training example for generating the first question in iterative fact-checking. Given a claim and associated metadata, the model must generate an initial, relevant fact-checking question without additional context.

Training Example for Subsequent Question Generation

Input:

```
Claim: Donald Trump delivered the largest tax cuts in
American history.
Metadata: {'claim_date': '25-8-2020', 'speaker':
'Eric Trump', 'original_claim_url': None,
'reporting_source': 'Speech at The Republican
National Convention', 'location_ISO_code': 'US'}
Context:
Q1: Did the 2017 tax bill deliver the largest tax cuts
in American history?
A1: This tax cut is the 8th largest as a percent of
Gross Domestic Product
(GDP) since 1918 and the 4th largest in inflation-
adjusted dollars.
Predict the next question:
```

Target:

```
Has there been a larger tax bill than the 2017 tax bill?
```

Figure 4: Example illustrating subsequent question generation. Given the claim, metadata, and previous question-answer pairs as context, the model generates the next relevant fact-checking question, building iteratively upon previously obtained information.

Training Example for Question Answering

Input:

Claim: The United States of America and its Western allies have been using their media outlets to publish articles based on fabricated information under allegations of non-compliance with the Chemical Weapons Convention.
Metadata: {'claim_date': '30-10-2020', 'speaker': 'Syrian Arab News Agency (SANA)', 'original_claim_url': 'https:...', 'reporting_source': 'Syrian state media outlet', 'location_ISO_code': 'SY'}
Evidence:
out by the Assad regime, usually dropped from the air, and Islamic State
[...]
basements, chlorine gas, which is heavier than air, sinks into these last refuges, finally forcing people to flee their homes and towns. Our research shows what Syrians on the ground have known for years: that chemical weapons have become a completely normalised component of the Syrian regime arsenal used for years in full view of the international community with near impunity, said Tobias Schneider, a GPPI research fellow who worked on the new resource. Syria is commonly described as the best documented war in
Question:
Has Syria complied with the Chemical Weapons Convention?
Answer the question and mention your source based on the provided evidence or metadata in JSON format:

Target:

{"source": "Evidence", "answer": "No"}

Figure 5: Example of structured training data used for question answering. The input contains a claim, associated metadata, and an evidence chunk retrieved based on semantic similarity. The target output specifies the answer to the fact-checking question and the source (evidence or metadata) from which the model derived this answer.

A.1.3 Training Example for Verdict Prediction

Figure 6 shows an example for Verdict Prediction.

Training Example for Veracity Prediction

Input:

Claim: Hunter Biden had no experience in Ukraine or in the energy sector when he joined the board of Burisma.
Metadata: {'claim_date': '25-8-2020', 'speaker': 'Pam Bondi', 'original_claim_url': None, 'reporting_source': 'Speech at The Republican National Convention', 'location_ISO_code': 'US'}
Context:
Q1: Did Hunter Biden have any experience in the energy sector at the time he joined the board of the Burisma energy company in 2014
A1: No
Q2: Did Hunter Biden have any experience in Ukraine at the time he joined the board of the Burisma energy company in 2014
A2: No
Write a justification and predict a label in JSON format:

Target:

{"justification": "No former experience stated.", "label": "Supported"}

Figure 6: Example of a structured training instance for veracity prediction. The input includes a claim, metadata, and previously answered question-answer pairs as context. The model must generate a concise justification and assign an appropriate veracity label (Supported, Refuted, Conflicting Evidence/Cherrypicking, or Not Enough Information (NEI)).