CHECKWHY: Causal Fact Verification via Argument Structure

Jiasheng Si[1,4†], **Yibo Zhao**[2,3†], **Yingjie Zhu**[2,3], **Haiyang Zhu**[2,3], **Wenpeng Lu**[1,4], **Deyu Zhou**[2,3*]
[1]Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center, Qilu University of Technology (Shandong Academy of Sciences), China
[2] School of Computer Science and Engineering, Southeast University, China
[3] Key Laboratory of New Generation Artificial Intelligence Technology and Its Interdisciplinary Applications (Southeast University), Ministry of Education, China
[4] Shandong Provincial Key Laboratory of Computer Networks, Shandong Fundamental Research Center for Computer Science, China
{jiashengsi, lwp}@qlu.edu.cn, {yibozhao, yj_zhu, haiyangzhu, d.zhou}@seu.edu.cn

Abstract

With the growing complexity of fact verification tasks, the concern with "thoughtful" reasoning capabilities is increasing. However, recent fact verification benchmarks mainly focus on checking a narrow scope of semantic factoids within claims and lack an explicit logical reasoning process. In this paper, we introduce CHECKWHY, a challenging dataset tailored to a novel causal fact verification task: checking the truthfulness of the causal relation within claims through rigorous reasoning steps. CHECKWHY consists of over 19K "*why*" *claim-evidence-argument structure* triplets with *supports*, *refutes*, and *not enough info* labels. Each argument structure is composed of connected evidence, representing the reasoning process that begins with foundational evidence and progresses toward claim establishment. Through extensive experiments on state-of-the-art models, we validate the importance of incorporating the argument structure for causal fact verification. Moreover, the automated and human evaluation of argument structure generation reveals the difficulty in producing satisfying argument structure by fine-tuned models or Chain-of-Thought prompted LLMs, leaving considerable room for future improvements[1].

1 Introduction

Fact verification is a crucial debunking task that entails verifying the truthfulness of claims by cross-referencing them with reliable evidence drawn from established resources (Guo et al., 2022b), which prevents the proliferation of erroneous information online and fosters public trust (Lewandowsky et al., 2020; Glockner et al., 2022). However, with the multi-step reasoning capability in fact verification models remaining uncertain (Schuster et al., 2019; Pan et al., 2023; Zhang et al., 2024), existing research reflects the deficiency in in-depth understanding of the explicit reasoning mechanisms when performing inference on multi-hop evidence. This prompts us to develop a strong benchmark that incorporates the interpretable "thought" process to assess the logical reasoning capabilities of models.

Currently, substantial progress has been made on common fact verification benchmarks, e.g., HOVER (Jiang et al., 2020), FEVEROUS (Aly et al., 2021), and SCITAB (Lu et al., 2023). Nevertheless, existing resources have inherent limitations. Classic datasets primarily focus on verifying the semantic factoids of "*who*", "*what*", "*when*", and "*where*" within the claim (Rani et al., 2023). For example, verifying the claim "*John Lennon was born before the astronaut who drank the first coffee in space.*" can be decomposed into verifying factoids such as "*who, and when the astronaut drank the first coffee in space.*" and "*when the man was born.*". However, these semantic factoids can be answered individually by straightforward factoids-matching between the claim and distinct independent evidence (Jiang et al., 2020; Pan et al., 2023), e.g., word overlapping or proof matching(Krishna et al., 2022), thereby limiting its significant potential to summarize the correlational evidence with "thought" steps. Heretofore, noticeably absent in prior datasets are "*why*" claims: containing causal relations that need to be verified. These claims prompt for not simple factoid matching, but an explicit logical reasoning path for verification (Ho et al., 2023).

More specifically, Figure 1 presents a "*why*" claim featuring a cause-effect pair where "*military crises*" causes the "*decrement of the purity of denarius silver.*". Verifying such causal relations is quite challenging, which necessitates complex logical reasoning and context information beyond

† Equal Contribution.
* Corresponding Author.
[1] The dataset and code are available at https://github.com/jasenchn/checkwhy

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 15636–15659
August 11-16, 2024

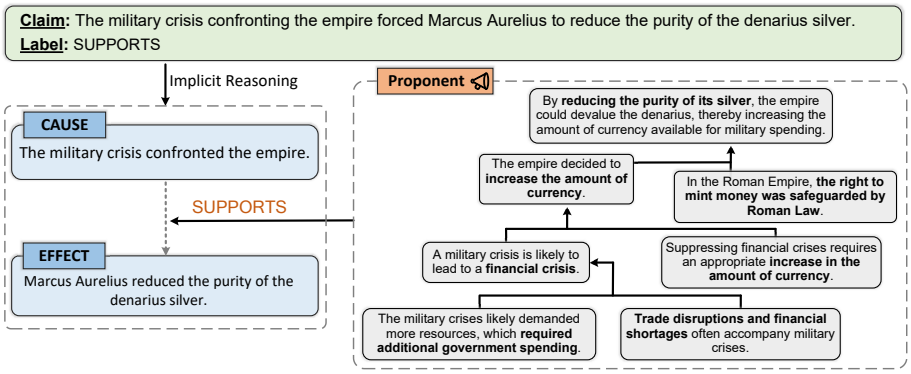

Figure 1: An entry from CHECKWHY: a "*why*" **claim** with its corresponding **cause** and **effect**, and an **argument structure** representing the reasoning process from cause to effect. Notably, the cause-effect pair is used solely during the annotation process and not included in the argument structure, implying that it is implicitly inferred from the claim, rather than being provided explicitly.

the factoids within the claim (Jin et al., 2023, 2024; Romanou et al., 2023). For instance, to *support* this causal relations (i.e., *military crises* $\xrightarrow{cause}$ *decrease purity of silver*), it is essential to incorporate the extra intermediate reasoning steps to bridge the connection between the cause-effect pair, thus forming a logical reasoning path: *military crises* $\xrightarrow{①}$ *financial crisis* $\xrightarrow{②}$ *increase amount of currency* $\xrightarrow{③}$ *decrease purity of silver*. The rationale behind ① is that *military crises* often coincide with extra factors such as *trade disruption* or *more government spending*, leading to the *financial crisis*. The reason supporting ② is that *increase amount of currency* is a demanding response to suppress the *financial crisis*. Furthermore, *the rights to mint money that is safeguarded by Roman Law* ensures the behavior of ③ is established. The above inference reveals a coherent reasoning process, which involves aligning the construction of a "thoughtful" logical structure among correlational evidence with causal relation verification.

In this paper, we introduce CHECKWHY, a challenging dataset built around a novel causal fact verification task: assessing whether the causal relation within the claim is valid by explicit logical structure. This dataset consists of 19,596 *claim-evidence-argument structure* triplets derived from the WIKIWHY dataset (Ho et al., 2023). The uniqueness of CHECKWHY is that each entry contains a "*why*" *claim* with causal relations and an *argument structure* formed by correlated evidence: the latter is inspired by the theory literature on argument structure (Grimshaw, 1990; Freeman, 2011), which depicts how the different statements fit together as wholes to allegedly lend support to the claim. Moreover, inspired by prior research (Glockner et al., 2021), we assume that the label of a causal relation within the claim depends on the provided argument structures, rather than the semantics itself. Thus, each claim is labeled as *supports*, *refutes*, or *not enough info* based on different argument structures. In addition, to prevent the bias in human cognition, we employ a human-model collaboration annotation approach to generate claims, evidence, and corresponding argument structures. Compared to existing datasets, CHECKWHY covers a variety of topics and argument structures, which may prove valuable for performing causal reasoning across various scenarios.

Based on the experiments on four tasks we propose and the human evaluation in our CHECKWHY, our experiments reveal the significance of incorporating the argument structure for causal fact verification. Meanwhile, our experiments in argument structure generation also validate the difficulty in producing satisfying argument structures for causal claims. Our key contributions are summarized as follows: (I) We propose verifying the "why" claims with causal relations through reasoning on argument structure as a novel causal fact verification formulation. (II) We construct CHECKWHY by introducing a human-model collaboration annotation approach, drawing inspiration from the theory research on argument structure. (III) We conduct thorough experiments on state-of-the-art models with four tasks, including fine-tuned models and LLMs, which investigates various settings and points out the potential for improvement.

2 Preliminaries

Our CHECKWHY framework is inspired by the argument structures theorized in the literature on

logical theory (Thomas, 1973; Toulmin, 2003; Walton et al., 2008; Walton, 2013). Thus, we begin by outlining the standard argumentation structure and then present a concise overview of our argument structure.

2.1 Standard Argumentation Structure

The standard argumentation structure (Thomas, 1973) is the scheme for structurally representing argument macrostructure: concerning what are the structural patterns in which the elements that constitute an argument may combine to form the overall argument. It consists of four basic structures.

• **Serial Argument** ($P \rightarrow C$): an argument structure has one premise P to give a reason to support the conclusion C.

• **Convergent Argument** ($P_1 \vee P_2 \rightarrow C$): an argument structure has more than one premise, with each one function separately ($P_1 \vee P_2$) as a reason to support the conclusion C.

• **Linked Argument** ($P_1 \wedge P_2 \rightarrow C$): an argument structure has more than one premise, and the premises function together ($P_1 \wedge P_2$) to give a reason to support the conclusion C.

• **Divergent Argument** ($P \rightarrow C_1/C_2$): an argument structure has more than one separate conclusion (C_1/C_2) that can be supported by the same premise P.

2.2 Our Argument Structure

The argument structure in CHECKWHY follows a tree-like framework, with the claim serving as the root node and evidence branching out as child nodes. These nodes are connected by directed edges that symbolize logical relations. Specifically, following the standard argumentation structure (Thomas, 1973), we blend various basic structures into a unified argument tree. This is achieved by referring to the semantics of each piece of evidence and the logical relations between different pieces of evidence. In this structure, the **series argument** leads from one child node to one parent node, while the **divergent argument** deduces multiple parent nodes from one child node. Furthermore, due to the challenging to discern the subtle distinction between *convergent* and *linked argument*, we merge these structural types into a new **combined argument**. This argument tree formalizes the reasoning process that begins with foundational evidence and progresses toward claim establishment.

A tricky issue here is that the argument structure inherently *supports* in the claim based on its definition, whereas the *refutes* instances are indispensable for causal fact verification. To tackle this, drawing inspiration from the *warrant* and *rebuttal* concepts in Toulmin's structure (Toulmin, 2003), wherein the warrant offers facts or rules to back up the claim, and the rebuttal indicates conditions that negate the claim. We apply diverse argument structures to the same claim to collect both *supports* and *refutes* instances. In specific, offering the basic argument structure (i.e., *warrant*) to acquire *support* labels, and providing another structure (i.e., *rebuttal*) that upholds the opposition of causal relation in the claim to obtain *refutes* labels.

3 The CHECKWHY Dataset

We adopt a human-model collaboration annotation approach to construct our CHECKWHY, as shown in Figure 2, which including data preparation (§ 3.1), generation with LLMs (§ 3.2), and manual validation (§ 3.3).

3.1 Data Preparation

We utilize the WIKIWHY dataset (Ho et al., 2023), a publicly available cause-and-effect QA resource, as our data source. Each QA pair in this dataset was annotated with effect (question), cause (answer), and associated explanations, making it well-suited for our purposes. From the entire collections of cause-effect pairs in WIKIWHY, we initially filter out instances that contain incomplete syntactic structure in either the cause or the effect, thus remaining cause-effect pairs with valid causal relations. This is realized by justifying the absence of a *predicate component* in the sentence using StanfordNLP Parser tool (Qi et al., 2020). To ensure quality, we include an option in the subsequent manual validation process to mark a claim as "Discard - not a valid causal relation, incomplete, or redundant". During this stage, we extract 7,403 cause-effect pairs from a total of 9,406 instances within the WIKIWHY dataset.

3.2 Generation with LLMs

The rich knowledge and generative capabilities of large language models (LLMs) raise widespread attention on their use in aiding the construction of dataset (Si et al., 2023; Chen et al., 2023). As such, we adopt GPT-4 (Achiam et al., 2023) for our data generation through in-context learning (Ouyang et al., 2022). Notably, for each cause-effect pair,

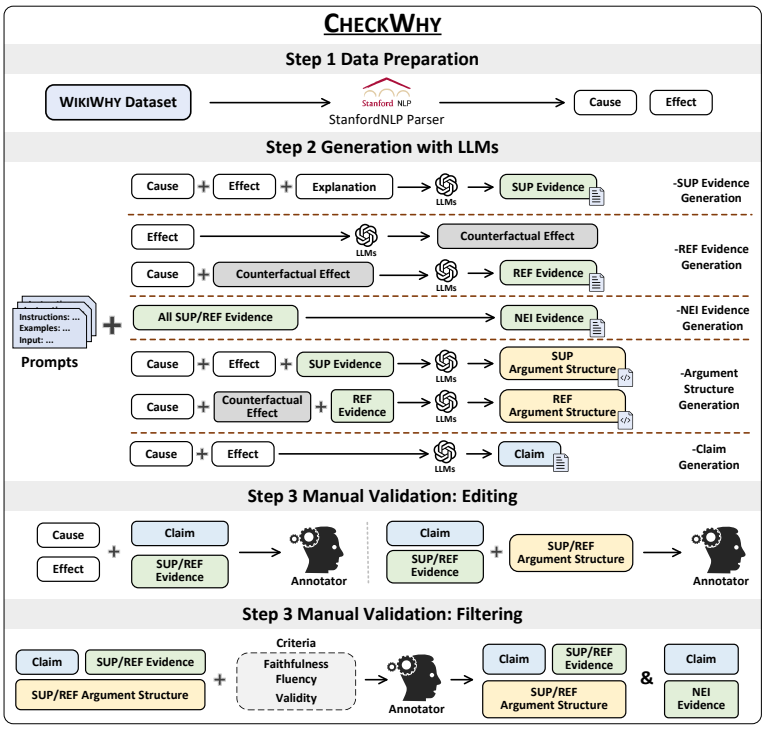

Figure 2: The human-model collaboration annotation process of CHECKWHY, which contains three steps: (I) data preparation; (II) generation with LLMs (including the generation of claim, evidence, and argument structure); (III) manual validation.

our CHECKWHY applies three argument structures to each claim to acquire the *SUP*, *REF*, and *NEI* instances with *supports*, *refutes*, and *not enough info* labels, which ensures a dataset with balanced distribution of veracity labels. All the prompts used in our annotation process are presented in Appendix E.

• **Support Evidence Generation** To obtain valid *SUP* instances, textual evidence that *support* the claim is generated via GPT-4. In specific, by taking the cause-effect pair and associated explanation within WIKIWHY as input, the GPT-4 is prompted with the designed `ReAct`-like (Yao et al., 2023) examples, enabling LLMs to logically deduce the effect from the cause through a step-by-step thought process. Finally, the reasoning process generated by LLMs is considered as *support* evidence.

• **Refute Evidence Generation** Due to the significantly broader generation scope compared to the *SUP* instances, building *REF* instances is a challenging task (Zhu et al., 2023; Tan et al., 2023). To produce valid evidence that refutes the claim, following Zhu et al. (2023), we apply the generation of evidence that refutes the counterfactual effect as an intermediate step. In specific, starting with the original cause, we prompt GPT-4 to generate the counterfactual effect by crafting an effect with an opposite label. Then, we prompt GPT-4 with `ReAct`-like examples to reason from cause to the counterfactual effect, resulting in the generation of evidence that refutes the original effect.

• **NEI Evidence Generation** Following Aly et al. (2021), we incorporate instances labeled as *not enough info* into our dataset. To this end, we either randomly eliminate evidence from the corresponding *SUP/REF* instances or merge evidence from these two types to generate *NEI* evidence.

• **Argument Structure Generation** To describe the logical reasoning process for each instance, we require GPT-4 to generate succinct argument structures for both the *SUP* and *REF* instances[2]. Inspired by Wang et al. (2023a), we prompt GPT-4 with Python-style code examples that have strict format restrictions. These examples help establish a logical reasoning path for each instance, thus deepening the structural understanding of arguments.

In specific, by referring to the argument structure outlined in § 2.2, we introduce the `FACT` class, which symbolizes an evidence node within the argument structure, and three distinct func-

[2] Since the containing of non-gold evidence for *NEI*, we abandon to generate the argument structure for *NEI*.

tions to regulate the generation of GPT-4. (I) The `one_to_one_linking` function (corresponding to *series argument*) represents the deduction from one piece of evidence to another. (II) The `one_to_multiple_linking` function (corresponding to *divergent argument*) illustrates the deduction from one piece of evidence to multiple pieces. (III) The `multiple_to_one_linking` function (corresponding to *combined argument*) represents the reasoning from several pieces of evidence to support a single one.

• **Claim Generation** By taking the cause-effect pairs extracted from WIKIWHY as input, the GPT-4 is prompted with crafted examples to generate diverse causal claims linguistically without accessing external information.

Notably, to better simulate the real-world fact verification scenarios and boost the distinguishing capability of models to the misleading evidence, we employ BM25 (Robertson and Zaragoza, 2009) to retrieve and extract 300 pieces of evidence from the entire dataset that show a higher degree of lexical overlap with the claim. Then, we use BLEURT (Sellam et al., 2020) to select 8-12 pieces of evidence with close semantic similarity to serve as distractor evidence for *SUP*, *REF*, and *NEI*.

3.3 Manual Validation

The uncontrollable generation of LLMs might result in unfaithful or invalid instances. Thus, we subsequently incorporate the human validation procedure for two purposes: (I) to revise and edit the content of generated instances by LLMs; (II)) to critically review and filter out the low-quality instances. The details are shown in Appendix C.

Manual Editing We ask a group of annotators with NLP backgrounds to edit and refine the generated instances.

• **Claim and Evidence** Annotators are required to review the claims and assess whether they accurately contain all details from the provided cause-effect pair and express clear causality. Then, annotators are instructed to refine the generated evidence by evaluating whether all evidence aligns with the label and is presented in concise language (e.g., without repetition or redundancy).

• **Argument Structure** The annotators are required to review and correct errors within argument structures by double-checking the validity of each reasoning step and removing unnecessary non-code content, such as comments. In this context, we encourage annotators to apply their judgment and expertise throughout the process.

Manual Filtering Another group of annotators is required to filter out the low-quality edited instances to further improve the quality of the dataset. Given the instructions and demonstration of the annotated argument structures, annotators need to decide whether to retain or discard instances based on the following three criteria. Finally, 6,757 and 6,638 instances pass the filter for *SUP* and *REF*, respectively.

• **Faithfulness:** The truthfulness of the claim can be verified by the evidence and is consistent with the given label.

• **Fluency:** The claim and associated evidence are written fluently and without grammatical errors.

• **Validity:** The argument structure is logically coherent and correctly depicts the reasoning process from cause to effect.

Quality Control We apply strict quality control measures by following the principles outlined in the Dataset Statement (Bender and Friedman, 2018), ensuring high-quality annotation. Specifically, for each instance, two annotators are required to perform the manual editing and filtering, and resolve any disagreements identified by a third annotator. We employ majority voting to determine the final retained instances, reaching an inter-rater agreement at Fleiss's $\kappa = 0.75$ (Fleiss et al., 1981). The details are listed in Appendix C.

4 Dataset Analysis

4.1 Dataset Statistics

Table 1 presents the statistics of our CHECKWHY dataset. This dataset comprises 6,532 sets of instances, with each set including a *SUP* instance, a *REF* instance, and an *NEI* instance, resulting in 19,596 instances in total. In addition, due to the lack of argument structure for *NEI* — the *SUP* and *REF* instances contain three elements {*claim*, *evidence*, *argument structure*}, while the *NEI* instance is limited to the *claim* and *evidence* — we build a CHECKWHY$_2$ dataset by excluding *NEI* instances to assess the reasoning ability through argument structures within *SUP*/*REF* instances. Compared to WIKIWHY, our CHECKWHY includes more extended and detailed reasoning steps, with an average of 13.7 pieces of evidence and 4.08 reasoning steps. The histograms of evidence length and reasoning step counts are detailed in Appendix B.

	CHECKWHY	CHECKWHY$_2$
# of All	19,596	13,062
# of Train	14,796	9,864
# of Dev	2,400	1,600
# of Test	2,400	1,600
Avg. # Evidence	11.1	13.7
Avg. # Distractor	-	9.5
Avg. # Tokens per Claim	24.3	24.3
Avg. # Tokens per Evidence	19.6	19.3
Avg. # Reasoning Steps	-	4.08

Table 1: Summary statistics of CHECKWHY. Here, CHECKWHY is the dataset containing {*SUP*, *REF*, *NEI*} instances, while CHECKWHY$_2$ is composed of {*SUP*, *REF*} instances. *Avg. #* denotes the number on average, and the *Evidence* denotes the combination of the valid annotated evidence and the extracted distractor evidence.

4.2 Argument Structure Analysis

Inspired by Dalvi et al. (2021), to understand the intricacies of reasoning present within CHECKWHY, we analyzed the reasoning steps extracted from 50 randomly selected instances with diverse argument structures. We identified 5 prevalent high-level categories of reasoning, as shown in Table 2. *Event causality* (36%) refers to the relations where one event directly leads to another event. *Inference from Properties* (22%) involves concluding by referring to the properties present within one input. *Rule-based Inference* (17%) is about deducing new information or making decisions grounded in a set of predefined rules or conditions. *Inductive reasoning* (14%) requires a model to make generalized conclusions based on specific observations or evidence, deriving general principles or patterns from particular examples or observations. Moreover, with 10% of instances require applying *sequential reasoning*. As a whole, this analysis reveals diverse forms of reasoning steps that are essential for inferring the argument structure in CHECKWHY. In addition, we summarize 5 types of prevalent complex argument structures, as shown in Figure 5, which also shows the complexity of our argument structure through a macro perspective.

5 Experiment

Task Notation The instance within CHECKWHY is denoted by the quadruple $(C, E/LE, S, Y_2|Y_3)$, where C denotes the claim, $E = \{e_1, e_2, \ldots, e_n\}$ denotes the evidence, LE refers to the initial leaf evidence in an argument structure. Notably, we ensure that LE includes the valid leaf evidence and all the distractor evidence, which requires the denoising capability of models. In addition, S denotes the argument structure, and $Y_3 \in \{REF, SUP, NEI\}$ or $Y_2 \in \{REF, SUP\}$ for CHECKWHY$_2$. Based on our CHECKWHY, we define four tasks of increasing difficulty, with the aim of (i) predicting whether the evidence supports or refutes the claim, or presents not enough information; (ii) generating valid argument structures outlining the reasoning process for causal fact verification. The following describes four tasks.

Task 1: Input = (C, LE)**, Output =** $Y_2|Y_3$. We follow the traditional fact verification formulation to evaluate the inference capability of models on the CHECKWHY dataset, i.e., verifying the claim based on foundational leaf evidence. In this vein, we conduct the experiments on three types of baselines: (I) Discriminative models: BERT (Devlin et al., 2019), Transformer-XH (Zhao et al., 2020), and UniXcoder (Guo et al., 2022a), (II) Generative models: CodeT5 (Wang et al., 2021) and CodeT5+ (Wang et al., 2023b), (III) LLMs: ChatGPT and GPT-4 [3](Achiam et al., 2023) with Chain-of-Thought prompts (Wei et al., 2022).

Task 2: Input = (C, LE, S)**, Output =** Y_2. Task 2 is designed to assess whether models have a better understanding when incorporating the structural information within the argument structure during the verification. In this vein, We adopt the same set of baselines as in Task 1.

Task 3: Input = (C, E)**, Output =** (Y_2, S). To investigate the logical reasoning ability of existing models when performing verification, we experiment that involves the simultaneous verification of the claim and the generation of the argument structure. Here, we conduct experiments on CodeT5, CodeT5+, GPT-4 and ChatGPT with Chain-of-Thought prompts.

Task 4: Input = (C, LE)**, Output =** (Y_2, S). Task 4 investigates whether the model can independently construct an entire argument structure from bottom to top based on leaf evidence and its reasoning ability, which is the most challenging experimental setup in this paper. The model must generate the verification label and a complete argument structure based on the input claim and leaf evidence nodes. We use the same baseline models as in Task 3 for this task.

[3] https://platform.openai.com/docs/models/gpt-4-and-gpt-4-turbo

Inference Type	Prop.		Example
Event causality	36%	E_1	Heavy rain and flooding have caused significant damage to the area.
		E_2	Road closures and traffic disruptions are affecting the entire city.
			$E_1 \rightarrow E_2$
Inference from Properties	22%	E_1	If the patient has a fever and cough, diagnose them with a respiratory infection.
		E_2	David has a fever and cough.
		E_3	he is likely suffering from a respiratory infection.
			$E_1, E_2 \rightarrow E_3$
Rule-based Inference	17%	E_1	All humans are mortal.
		E_2	Socrates is a human.
		E_3	Therefore, Socrates is mortal.
			$E_1, E_2 \rightarrow E_3$
Inductive reasoning	14%	E_1	Every morning for the past week, you've woken up to find the grass wet.
		E_2	Based on these observations, you hypothesize that it rains during the night.
			$E_1 \rightarrow E_2$
Sequential reasoning	10%	E_1	A heavy rainstorm has occurred in the capital of Italy.
		E_2	The capital of Italy is Rome.
		E_3	Rome has experienced a heavy rainstorm.
			$E_1, E_2 \rightarrow E_3$

Table 2: The prevalence of 5 reasoning steps required for reasoning on argument structure, sampled from 50 random instances in the training set. Here, E_n denotes input evidence, and the "$\rightarrow$" symbolizes the "reasoning".

Model	Task1 (Y_3)		Task1 (Y_2)		Task2	
	Acc.	F1	Acc.	F1	Acc.	F1
Discriminative Models						
BERT (FT)	**73.3**	**72.9**	76.9	76.9	88.0	88.0
Transformer-XH (FT)	63.2	62.2	**78.9**	**78.9**	**90.4**	**90.4**
UniXcoder (FT)	70.1	69.5	77.2	77.2	83.4	83.4
Generative Models						
CodeT5 (FT)	72.3	72.0	71.6	71.6	78.0	78.0
CodeT5+ (FT)	72.8	72.5	65.6	65.4	79.3	79.3
Large Language Models						
ChatGPT (CoT)	37.8	31.1	60.6	60.7	64.4	60.7
GPT-4 (CoT)	49.6	41.0	62.5	62.3	72.8	71.0

Table 3: The performance of baselines on Task 1 and Task 2, where FT denotes that the model is fine-tuned on CHECKWHY training set. The CoT denotes the Chain-of-Thought prompts with few examples. The best results are marked in **bold** and the results with further improvement after the incorporation of argument structures are underlined.

5.1 Evaluation Metrics

Automatic Evaluation Metircs Due to the uniqueness of argument structure within CHECKWHY, traditional evaluation metrics in text generation may not be suitable in our setting. Thus, we propose two new evaluation metrics focusing on argument structure generation.

• **Structure Similarity** Inspired by Saha et al. (2021), we introduce an automated evaluation metric named Structure Similarity. The metric treats the argument structure as a collection of edges, with each edge considered as a sentence, and a matching method is employed to determine the optimal alignment between the edges in the predicted structure and those in the gold argument structure. In our experiments, the BERTScore based on DeBERTa-large (He et al., 2021) is used as the scoring function to measure how closely the predicted edges match the gold edges. The detailed algorithm can be found in Saha et al. (2021).

• **Exact Match Similarity** Similar to structure similarity, exact match similarity employs a stricter matching strategy. Here, structures are also regarded as sequences of edges, but a match is considered successful only when the predicted edge exactly matches the gold edge.

Human Evaluation Criteria To improve the reliability of the evaluation on argument structure, we introduce extra human evaluations beyond the automated evaluation metircs. Empirically, we randomly select 50 instances and ask three graduate students with NLP background to assess the generated argument structure according to the following criteria with binary score.

• **Validity:** Is the generated argument structure logically coherent?

• **Win/Tie/Lose:** Comparing the generated argument structure against the provided reference. Mark *Win* if you prefer the generated structure, *Tie* if you have no preference, and *Lose* if you prefer the reference structure.

5.2 Results and Discussion

Main Results The overall results of baselines on our CHECKWHY dataset are reported in Ta-

	Model	Automatic		Human				Verification	
		SS (↑)	ES (↑)	Win (↑)	Tie	Lose (↓)	Validity (↑)	Acc.	F1
Task 3	***Generative Models***								
	CodeT5 (FT)	87.4	48.4	10.0	27.5	62.5	52.5	83.7	83.5
	CodeT5+ (FT)	**88.8**	**53.4**	7.5	40.0	52.5	**62.5**	**86.6**	**86.6**
	Large Language Models								
	ChatGPT (CoT)	69.5	26.4	0.0	17.5	82.5	37.5	63.4	58.5
	GPT-4 (CoT)	77.7	34.2	**21.6**	**40.5**	**37.8**	62.2	72.5	70.9
Task 4	***Generative Models***								
	CodeT5 (FT)	72.8	1.1	5.5	30.3	64.2	55.2	67.9	68.9
	CodeT5+ (FT)	**75.7**	**2.5**	6.1	**33.9**	**60.0**	**59.5**	**72.6**	**72.6**
	Large Language Models								
	ChatGPT (CoT)	60.9	1.8	4.2	28.5	67.3	52.9	53.9	44.7
	GPT-4 (CoT)	66.7	1.4	**6.2**	33.2	60.6	57.8	57.0	47.8

Table 4: The performance of baselines on Task 3 and Task 4, where SS denotes structure similarity and ES denotes exact match similarity. The best results are marked in **bold**.

ble 3 and Table 4. In general, for Task 1 and Task 2, generative models show notably poorer performance compared to the discriminative models, and LLMs encounter substantial challenges in causal verification. Despite the powerful capability of GPT-4, which shows an improvement over ChatGPT for verification, it still lags behind the top-performing fine-tuned models. In addition, through the comparison of Y_3 and Y_2 on Task 1, we observe the consistent improvement for both discriminative models and LLMs, which indicates the disruptive impact brought by the ambiguous evidence within *NEI* instance, particularly on LLMs. An interesting observation here is that there is a slight decrease in the performance of generative models, which may be attributed to the disparities between the datasets utilized during the pretraining and fine-tuning phases. For Task 3 and Task 4, compared to the generative models, a similar phenomenon observed is that LLMs struggle to achieve satisfactory results in the generation of argument structure and show notably poorer performance on causal verification. This may be due to the conflict raised by the internal world knowledge and the evidence we generated, e.g., there is evidence that contradicts the real-world situation. In addition, the difference can be observed in the human evaluation, where LLMs present a slightly higher performance compared to the generative models. This may be attributed to the fact that LLMs generate more fluency and human-readable text with the powerful generative capability. Overall, the results of four tasks validate the difficulty when facing the setting of our CHECKWHY.

The Effectiveness of Argument Structures Argument structures prove to be particularly advantageous, especially in specific experimental setups. In Task 1, where structures are absent, models such as the CodeT5 series and LLMs, relying solely on textual information, may struggle to achieve high accuracy. Furthermore, when comparing Task 1 and Task 2, a consistent improvement is noted across all baselines if structures are directly provided. For example, GPT-4 achieves approximately a 10% increase in accuracy in Task 2 compared to Task 1. This effect shows the most prominent performance on Transformer-XH, a GNN method conducting reasoning over the evidence graph. This underscores the significance of structural information within the argument structure for causal fact verification.

The Construction of Argument Structures In Task 3 and Task 4, we investigate whether models can generate argument structures. Structure similarity may be significantly higher than exact match similarity because structure similarity considers the semantic meaning within nodes, whereas exact match similarity only checks whether characters are identical. CodeT5+ exhibits particularly noticeable performance, achieving the highest exact match similarity and structure similarity, as well as the highest F1 score and accuracy. Task 4 investigates whether models can construct an entire structure and complete inference given only the leaf nodes. All models perform poorly on Task 4, which may be attributed to the vast generation space in creating an entire structure. This indicates that when

all evidence nodes are provided, the model only needs to connect these nodes, instead of inferring the complete reasoning path. However, when only leaf nodes are provided, the task becomes much more challenging.

Human Evaluation Our human evaluation experiments, as detailed in Table 4, reveal that there is significant room for improvement across different aspects. None of the baseline models, including CodeT5, CodeT5+, and GPT-4, exhibit notably superior performance. Specifically, our strongest baseline, CodeT5+, generates valid argument structures only 62.5% of the time in Task 3, whereas in Task 4 it generates up to 60.0% of argument structures that are worse than the gold references. These results from the strongest LLMs leave ample room for improvement and serve as motivation for future work on these tasks.

6 Related work

6.1 Fact Verification Datasets

The fact verification task has recently garnered significant attention within the NLP community. Researchers prompt the development of fact verification by introducing various resources. Well-known datasets such as FEVER (Thorne et al., 2018), PolitiFact (Garg and Sharma, 2020), and CREAK (Onoe et al., 2021) regard the fact verification task as a form of natural language inference, aiming to predict whether the evidence supports or contradicts a claim. Subsequently, some datasets have been proposed to address fact-checking for complex claims necessitating multi-step reasoning (Aly et al., 2021; Jiang et al., 2020), which typically present multiple pieces of evidence linked to the claim through entity matching. Additionally, to improve the interpretability of the veracity predictions, Alhindi et al. (2018) extends the LIAR dataset by incorporating summaries from fact-checking articles. Furthermore, Kotonya and Toni (2020) introduces the first dataset explicitly containing gold explanations, composed of fact-checking articles and other news items. Rani et al. (2023) leverages interrogative questions to enhance interpretability by exploring the semantics factoids within the claim

Despite the advancement, existing datasets are primarily designed for verifying the semantic factoids with limited types within claims. Our work for the first time focuses on "why" claims: verifying the causal relation within the claim rather than basic semantic factoids. In addition, inspired by the literature on logical theory, we construct argument structures to explicitly represent the reasoning process, enhancing the causal reasoning ability and interpretability of the model.

6.2 Causal Reasoning

With the increasing attention on causality (Willig et al., 2023; Zhang et al., 2023), several formulations of causality-related skills for NLP are proposed, which can be summarized into (1) causality as knowledge (Sap et al., 2019), encompasses the representation, understanding, and utilization of causal relationships embedded within textual data, (2) causality as language comprehension (Stede, 2008; Cao et al., 2022; Yu et al., 2019), originates from traditional linguistic studies on causal connectives and the usage of causal language, extending to more recent efforts in causal relation extraction, and (3) causality as reasoning (Jin et al., 2023), involves identifying factors leading to outcomes and understanding underlying mechanisms. Our work is the first attempt to introduce causality into the fact verification task inspired by the causal reasoning in argumentation (Habernal et al., 2018; Heinisch et al., 2022), in which causal knowledge, causal language comprehension, and causal reasoning are significant. Exploring whether models can grasp the causal relationship between evidence and claim in the verification process will further promote the development of the task.

7 Conclusion

In this paper, we introduce CHECKWHY, a challenging dataset and benchmark built around a novel causal fact verification task, annotated by a human-model collaborative approach. By drawing inspiration from the logical theory, we incorporate the argument structure to represent the explicit logical reasoning process when assessing the causal relation within the "why" claim, which may prove valuable for developing multi-step reasoning skills across various scenarios. Extensive experiments on various state-of-the-art models validate the importance of incorporating argument structures and the difficulty of generating them. With the baselines achieving limited results, we believe that CHECKWHY is a challenging yet attractive benchmark for the development of fact verification systems.

Limitations

• Within the CHECKWHY framework, the label assigned to each claim is based on distinct argument structures. Therefore, the assigned label may not always correspond to real-world circumstances.
• The CHECKWHY dataset is created by LLMs, which could face difficulties in retrieving evidence in open-domain scenarios compared to previous datasets. This includes selecting relevant evidence in a broad and unrestricted information space.
• The initial evidence in the argument structure, as provided by LLMs, may not meet everyone's expectations or align with their understanding. This could affect the universal acceptability and perceived validity of the evidence.

Acknowledgement

The authors would like to thank the anonymous reviewers for their insightful comments. This work is funded by the National Natural Science Foundation of China (Grant No.62176053, No.62376130), Shandong Provincial Natural Science Foundation (Grant No.ZR2022MF243), Program of New Twenty Policies for Universities of Jinan (Grant No.202333008), and supported by the Big Data Computing Center of Southeast University.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Tariq Alhindi, Savvas Petridis, and Smaranda Muresan. 2018. Where is your evidence: Improving fact-checking by justification modeling. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 85–90.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*.

Emily M. Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Angela Cao, Gregor Williamson, and Jinho D. Choi. 2022. A cognitive approach to annotating causal constructions in a cross-genre corpus. In *Proceedings of the Linguistic Annotation Workshop (LAW-XVI)*, pages 151–159.

Zeming Chen, Qiyue Gao, Antoine Bosselut, Ashish Sabharwal, and Kyle Richardson. 2023. DISCO: Distilling counterfactuals with large language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 5514–5528.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. 2021. Explaining answers with entailment trees. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 7358–7370.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186.

Joseph L Fleiss, Bruce Levin, Myunghee Cho Paik, et al. 1981. The measurement of interrater agreement. *Statistical methods for rates and proportions*, 2(212-236):22–23.

James B Freeman. 2011. *Argument Structure:: Representation and Theory*. Springer Science & Business Media.

Sonal Garg and Dilip Kumar Sharma. 2020. New politifact: A dataset for counterfeit news. In *International Conference System Modeling and Advancement in Research Trends (SMART)*, pages 17–22.

Max Glockner, Yufang Hou, and Iryna Gurevych. 2022. Missing counter-evidence renders NLP fact-checking unrealistic for misinformation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 5916–5936.

Max Glockner, Ieva Staliunaite, James Thorne, Gisela Vallejo, Andreas Vlachos, and Iryna Gurevych. 2021. Ambifc: Fact-checking ambiguous claims with evidence. *Transactions of the Association for Computational Linguistics*, 12:1–18.

Jane Grimshaw. 1990. *Argument structure.* the MIT Press.

Daya Guo, Shuai Lu, Nan Duan, Yanlin Wang, Ming Zhou, and Jian Yin. 2022a. UniXcoder: Unified cross-modal pre-training for code representation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 7212–7225.

Zhijiang Guo, Michael Sejr Schlichtkrull, and Andreas Vlachos. 2022b. A survey on automated fact-checking. *Trans. Assoc. Comput. Linguistics*, pages 178–206.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. 2018. The argument reasoning comprehension task: Identification and reconstruction of implicit warrants. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1930–1940.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *Proceedings of the International Conference on Learning Representations*.

Philipp Heinisch, Anette Frank, Juri Opitz, Moritz Plenz, and Philipp Cimiano. 2022. Overview of the 2022 validity and novelty prediction shared task. In *Proceedings of the Workshop on Argument Mining*, pages 84–94.

Matthew Ho, Aditya Sharma, Justin Chang, Michael Saxon, Sharon Levy, Yujie Lu, and William Yang Wang. 2023. Wikiwhy: Answering and explaining cause-and-effect questions. In *Proceedings of the International Conference on Learning Representations*.

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*, pages 3441–3460.

Zhijing Jin, Yuen Chen, Felix Leeb, Luigi Gresele, Ojasv Kamal, Zhiheng LYU, Kevin Blin, Fernando Gonzalez Adauto, Max Kleiman-Weiner, Mrinmaya Sachan, and Bernhard Schölkopf. 2023. Cladder: Assessing causal reasoning in language models. In *Advances in Neural Information Processing Systems*.

Zhijing Jin, Jiarui Liu, Zhiheng Lyu, Spencer Poff, Mrinmaya Sachan, Rada Mihalcea, Mona Diab, and Bernhard Schölkopf. 2024. Can large language models infer causation from correlation? In *Proceedings of the International Conference on Learning Representations*.

Neema Kotonya and Francesca Toni. 2020. Explainable automated fact-checking for public health claims. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing*, pages 7740–7754.

Amrith Krishna, Sebastian Riedel, and Andreas Vlachos. 2022. Proofver: Natural logic theorem proving for fact verification. *Transactions of the Association for Computational Linguistics*, 10:1013–1030.

Stephan Lewandowsky, John Cook, UKH Ecker, D Albarracín, MA Amazeen, P Kendeou, D Lombardi, EJ Newman, G Pennycook, E Porter, et al. 2020. The debunking handbook. *The Debunking Handbook*, pages 13–29.

Xinyuan Lu, Liangming Pan, Qian Liu, Preslav Nakov, and Min-Yen Kan. 2023. SCITAB: A challenging benchmark for compositional reasoning and claim verification on scientific tables. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 7787–7813.

Yasumasa Onoe, Michael Zhang, Eunsol Choi, and Greg Durrett. 2021. Creak: A dataset for commonsense reasoning over entity knowledge. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744.

Liangming Pan, Xinyuan Lu, Min-Yen Kan, and Preslav Nakov. 2023. Qacheck: A demonstration system for question-guided multi-hop fact-checking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 264–273.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 101–108.

Anku Rani, S.M Towhidul Islam Tonmoy, Dwip Dalal, Shreya Gautam, Megha Chakraborty, Aman Chadha, Amit Sheth, and Amitava Das. 2023. FACTIFY-5WQA: 5W aspect-based fact verification through question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 10421–10440.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Angelika Romanou, Syrielle Montariol, Debjit Paul, Leo Laugier, Karl Aberer, and Antoine Bosselut. 2023. CRAB: Assessing the strength of causal relationships between real-world events. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 15198–15216.

Swarnadeep Saha, Prateek Yadav, Lisa Bauer, and Mohit Bansal. 2021. ExplaGraphs: An explanation graph generation task for structured commonsense reasoning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 7716–7740.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019.

Atomic: an atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence and Innovative Applications of Artificial Intelligence Conference*, pages 3027–3035.

Tal Schuster, Darsh J. Shah, Yun Jie Serene Yeo, Daniel Filizzola, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 3417–3423.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892.

Chenglei Si, Navita Goyal, Sherry Tongshuang Wu, Chen Zhao, Shi Feng, Hal Daumé III, and Jordan Boyd-Graber. 2023. Large language models help humans verify truthfulness–except when they are convincingly wrong. *arXiv preprint arXiv:2310.12558*.

Manfred Stede. 2008. Connective-based local coherence analysis: A lexicon for recognizing causal relationships. In *Semantics in Text Processing. STEP Conference Proceedings*, pages 221–237.

Neset Tan, Trung Nguyen, Josh Bensemann, Alex Peng, Qiming Bao, Yang Chen, Mark Gahegan, and Michael Witbrock. 2023. Multi2Claim: Generating scientific claims from multi-choice questions for scientific fact-checking. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, pages 2652–2664.

Stephen Naylor Thomas. 1973. *Practical Reasoning in Natural Language*. Prentice-Hall.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 809–819.

Stephen E Toulmin. 2003. *The uses of argument*. Cambridge University Press.

Douglas Walton. 2013. *Argumentation schemes for presumptive reasoning*. Routledge.

Douglas Walton, Christopher Reed, and Fabrizio Macagno. 2008. *Argumentation schemes*. Cambridge University Press.

Xingyao Wang, Sha Li, and Heng Ji. 2023a. Code4Struct: Code generation for few-shot event structure prediction. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 3640–3663.

Yue Wang, Hung Le, Akhilesh Gotmare, Nghi Bui, Junnan Li, and Steven Hoi. 2023b. CodeT5+: Open code large language models for code understanding and generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1069–1088.

Yue Wang, Weishi Wang, Shafiq Joty, and Steven C.H. Hoi. 2021. CodeT5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 8696–8708.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Moritz Willig, Matej Zečević, Devendra Singh Dhami, and Kristian Kersting. 2023. Probing for correlations of causal facts: Large language models and causality.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *Proceedings of the International Conference on Learning Representations*.

Bei Yu, Yingya Li, and Jun Wang. 2019. Detecting causal language use in science findings. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 4664–4674.

Cheng Zhang, Stefan Bauer, Paul Bennett, Jiangfeng Gao, Wenbo Gong, Agrin Hilmkil, Joel Jennings, Chao Ma, Tom Minka, Nick Pawlowski, et al. 2023. Understanding causality with large language models: Feasibility and opportunities. *arXiv preprint arXiv:2304.05524*.

Congzhi Zhang, Linhai Zhang, and Deyu Zhou. 2024. Causal walk: Debiasing multi-hop fact verification with front-door adjustment. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul N. Bennett, and Saurabh Tiwary. 2020. Transformer-xh: Multi-evidence reasoning with extra hop attention. In *Proceedings of the International Conference on Learning Representations*.

Yingjie Zhu, Jiasheng Si, Yibo Zhao, Haiyang Zhu, Deyu Zhou, and Yulan He. 2023. EXPLAIN, EDIT, GENERATE: Rationale-sensitive counterfactual data augmentation for multi-hop fact verification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 13377–13392.

A Experiment Details

In the fine-tuning process for Discriminative Models, we choose BERT-base-uncased, Transformer-XH (using BERT-base-uncased as the backbone), and UniXcoder-base. The batch size is set to 8, and we utilize the AdamW optimizer with a learning rate of 5e-6.

We select CodeT5-base and CodeT5+ (0.7B) for fine-tuning generation models, with a batch size of 4 and a learning rate of 1e-5. Additionally, we specify a maximum input length of 512 and a maximum generation length of 400.

Regarding Large Language Models (LLMs), our choices are gpt-4-1106-preview and ChatGPT-turbo, with a temperature setting of 0.1.

B Ablation Experiment

Number of Irrelevant Evidence We perform experiments on Task 4 with different average amounts of irrelevant evidence. The findings are summarized in Table 5, suggesting that task difficulty increases slightly as the number of irrelevant evidence increases.

Number of Reasoning Steps We group the results of Task 4 according to the number of reasoning steps in the gold argument structures. The outcomes are outlined in Table 6, demonstrating a significant decrease in scores as the number of steps increases.

Further Evaluation We refer to the evaluation metrics from EntailmentBank and make slight modifications to further evaluate the argument structures generated in Task 3 and Task 4. The specific results are shown in the Table 8.

• Leaf Nodes (F1, AllCorrect): Does the predicted argument structure use the correct leaf evidence? We compute an F1 score by comparing predicted leaf evidence to golden leaf evidence. The "AllCorrect" score is 1 if all nodes are identified correctly (F1=1.0), 0 otherwise.

• Steps (F1, AllCorrect): Are the individual reasoning steps structurally correct? As each intermediate node represents (the conclusion of) a single step, the step is considered structurally correct (score 1) if it perfectly matches the gold, 0 otherwise. We then measure F1 comparing all steps in the two trees. Then AllCorrect=1 if F1=1.0, 0 otherwise.

• Intermediates (F1, AllCorrect): Are the intermediate conclusions correct? For comparing gold and generated conclusions. F1 is computed using the number of aligned, correct intermediates wrt. the number of gold/predicted intermediates. AllCorrect=1 if F1=1, otherwise 0.

Based on the experimental results, the performance of the fine-tuned model significantly surpasses that of large language models (LLMs). From the Leaves metric, it is evident that the model can identify most of the evidence related to verification. However, the last two metrics indicate that the model encounters difficulties in linking these pieces of evidence to form an argumentative structure.

Pipeline Setup We also design an experiment in pipeline mode, using both CodeT5 and CodeT5+ as baseline models. In the first stage, modles output argument structures, and in the second stage, they produce the final classification results. The experimental results shown in Table 7 demonstrate that CodeT5+ generates better argument structures, and achieves better classification performance.

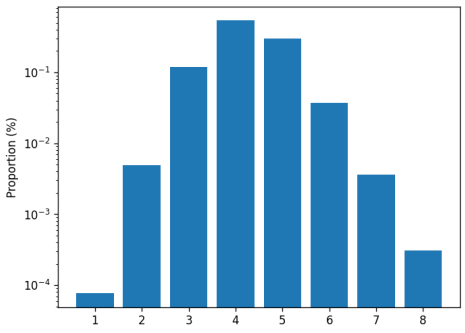

Figure 3: Histogram of evidence number in the CheckWhy

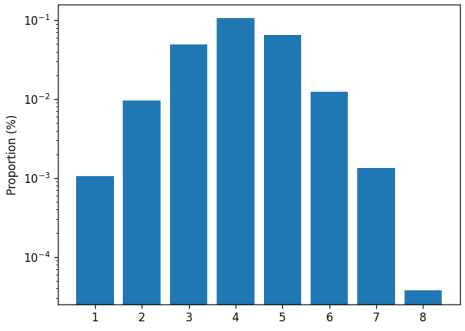

Figure 4: Histogram of argument steps in the CheckWhy

Number of Irrelevant Evidence	Acc	F1	SS	ES
6	67.6	66.8	72.3	1.3
8	69.3	69.1	75.7	2.5
10	65.4	63.9	71.9	1.0
12	62.3	60.0	71.0	1.5

Table 5: The performance of CodeT5 on Task 4 involving different numbers of irrelevant evidence.

Number of Steps	Number of instances	Acc	F1	SS	ES
1	40	80	80	78.6	8.6
2	273	74.3	72	77.4	2.6
3	734	68.9	66.4	73.3	0.7
4	464	65.7	65.4	70.4	0.1
5	77	64.9	59.6	64.6	0.3
≥ 6	12	58.3	49.6	62.7	0

Table 6: Results on Task 4 with the varying number of argument steps in the gold structure.

C Detailed Description of the Annotation Process

Source of Annotator To ensure the quality of our complex annotation process, we do not adopt CrowdSource platforms, even though more CrowdWorkers can be employed. We hire 20 university students with experience in NLP, especially those majoring in logical reasoning or argument mining. Before the formal annotation, we draft detailed guidelines, set up the annotation platform, and conduct three rounds of thorough training and one round of testing. In this way, 13 students pass the exam and are retained. Furthermore, we split these students into two groups, where 1 PhD student and 8 postgraduate students perform the Editing, and 1 PhD student and 3 postgraduate students perform the Filtering. Moreover, we apply a strict quality control procedure during our annotation process. In addition, authors conduct casual inspections during the annotation process and rate each annotator, with annotators who receive low scores undergoing additional training.

Annotation Cost The annotating time for each sample ranges from 8 to 30 minutes, averaging 15 minutes per sample. After annotation, we follow local labor laws, and each annotator is paid $3.05 per hour.

Due to the complex argument structure within each instance, the analysis of argument structure type and evaluation of prediction is quite time-consuming and labor-sensitive, especially facing complex structures. The average time for analyzing argument structure type is more than 20 minutes per instance, and the average time of evaluation on prediction ranges from 15 minutes to 20 minutes per instance, depending on different baseline models.

Model	Acc	F1	SS	ES
CodeT5	71.1	70.7	72.9	1.4
CodeT5+	75.9	76.1	76.3	2.6

Table 7: Results on Task 4 with the pipeline setting.

Quality Contol We implemented several quality control measures to minimize the possibility of annotators cutting corners and to ensure the quality of the data:

• **Editing** Given each instance, two annotators are required to perform manual editing of the Claim and Evidence and to resolve any errors identified by the third annotator. The final result is chosen by voting from all three annotators. Due to the complexity of the argument structure and its highly time-consuming nature, we don't apply this procedure to the editing of the argument structure.

• **Filtering** We apply a strict quality control procedure by filtering each instance with three anonymous annotators, wherein two annotators are required to filter the instance, and the third annotator (the Ph.D. student) serves as the supervisor. The decision of each instance is made by all three annotators. Moreover, the data transfer is completed by our annotation platform. According to our statistics, the ratio of "all-pass" is acceptable. A display of an example modified by humans is shown in Table 10.

• **Automatic Check** We have implemented a grammar check mechanism(e.g. the argument graph must be a connected graph, etc.) to verify the legality of the annotated argument structures. Before annotators can submit their results, they must pass this check.

• **Human Check** The authors conduct a random casual inspection of the edited data and rate each annotator. Moreover, since we have separated the annotators into two groups at different stages if the annotators in Filtering find low-quality edited instances, the annotators in Editing will receive a low rating.

D Annotation Interface

Figure 8 to Figure 11 illustrate the interfaces utilized for claim annotation, evidence annotation, argument map annotation, and the final review stage.

	Model	Automatic		Human		Verification	
		F1	AllCorrect	F1	AllCorrect	F1	AllCorrect
Task 3	***Generative Models***						
	CodeT5 (FT)	**85.6**	**60.8**	**20.7**	**3.4**	36.1	4.4
	CodeT5+ (FT)	84.4	59.3	20.5	**3.4**	**36.6**	**6.3**
	Large Language Models						
	GPT-4 (CoT)	51.2	12.4	9.1	0.2	14.2	1.42
	ChatGPT (CoT)	43.2	16.4	10.0	0	14.0	0.6
Task 4	***Generative Models***						
	CodeT5 (FT)	**81.1**	46.3	9.2	0.06	**12.2**	0.06
	CodeT5+ (FT)	77.8	42.6	15.4	**8.7**	9.4	0.06
	Large Language Models						
	GPT-4 (CoT)	79.7	**53.4**	**16.4**	7.2	8.16	0.2
	ChatGPT (CoT)	61.3	32.7	15.5	3.7	7.6	**0.4**

Table 8: The results evaluated using the metrics from the ENTAILMENTBANK (Dalvi et al., 2021).

Dataset	Reasoning Structure	Complicated Multi-hop	Causal Verification
FEVEROUS	✗	✓	✗
FEVER	✗	✗	✗
HOVER	✗	✓	✗
CHECKWHY	✓	✓	✓

Table 9: Comparison of CHECKWHY with previous Fact Verification datasets

These figures include comprehensive instructions, as well as contextual information for cause-effect pairs, to aid annotators in their understanding. Additionally, examples for each annotation stage are provided to enhance annotator proficiency.

E Generation Prompts

The prompts utilized during the data collection stage are depicted in Table 11 to Table 14. Meanwhile, the prompts for experiments can be found in Table 15 to Table 18.

Claim: The existence of an ancient lake thousands of years ago has led to the flat geography of South Jordan today.
GPT-4: SUPPORT Evidence
G1: Ancient lakes, when dried up, leave behind sediment that is evenly spread out, resulting in flat land.
G2: Geological evidence suggests that South Jordan was once covered by an ancient lake that left sediment deposits.
G3: The sediment from the ancient lake in South Jordan has, over thousands of years, compacted and created a flat landscape.
G4: No significant geological events have occurred to disrupt the flatness of the ancient lake bed in South Jordan.
G5: The ancient lake's flat bed is the primary reason for South Jordan's flatness, as no other events have altered the landscape.
GPT4: Argument Structure
G2 → (G1, G3) → G5
(G4, G5) → Claim
Human: SUPPORT Evidence
E1: Ancient lakes, when dried up, leave behind sediment that is evenly spread out~~,resulting in flat land~~.
E2: Geological evidence suggests that South Jordan was once covered by an ancient lake~~that left sediment deposits~~.
E3: In the South Jordan area, dried-up lakes leave behind sediment that is evenly distributed.
E4: Sediments from the ancient lake in South Jordan have been **compacted** over thousands of years~~, compacted and created a flat landscape~~.
E5: No significant geological events have occurred to disrupt the ~~flatness~~ **landscape** of the ancient lake bed in South Jordan.
~~G5: The ancient lake's flat bed is the primary reason for South Jordan's flatness, as no other events have altered the landscape.~~
E6: **The compacted sediment may eventually form a flat landscape.**
Human: Argument Structure
(E1, E2) → E3 → E4 → E6
(E5, E6) → Claim

Table 10: A comparison example between texts written by GPT-4 and human modifications.

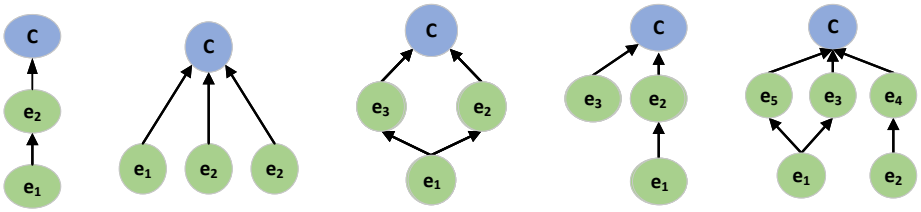

Figure 5: Types of Argument Structure

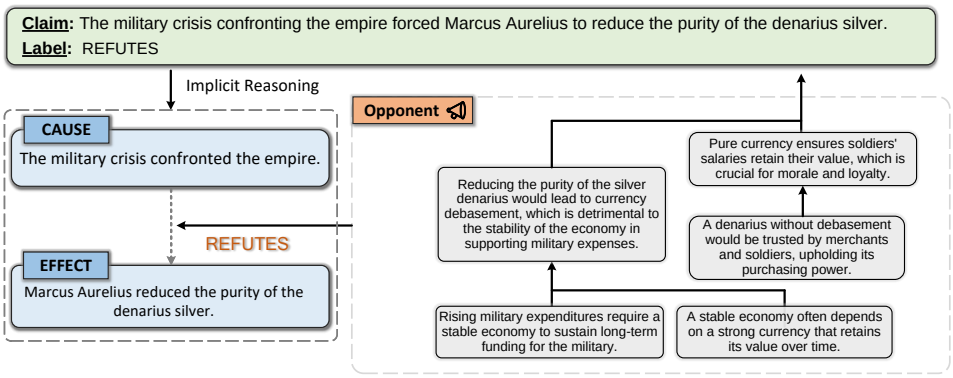

Figure 6: An entry with *REFUTES* labels from CHECKWHY

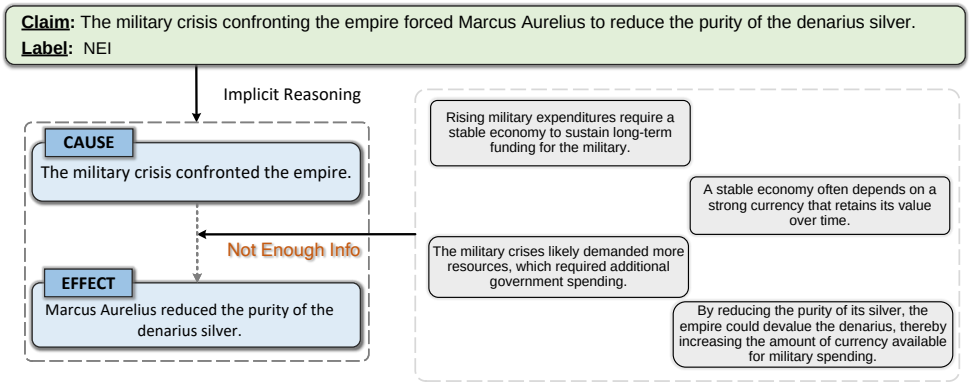

Figure 7: An entry with *NEI* labels from CHECKWHY

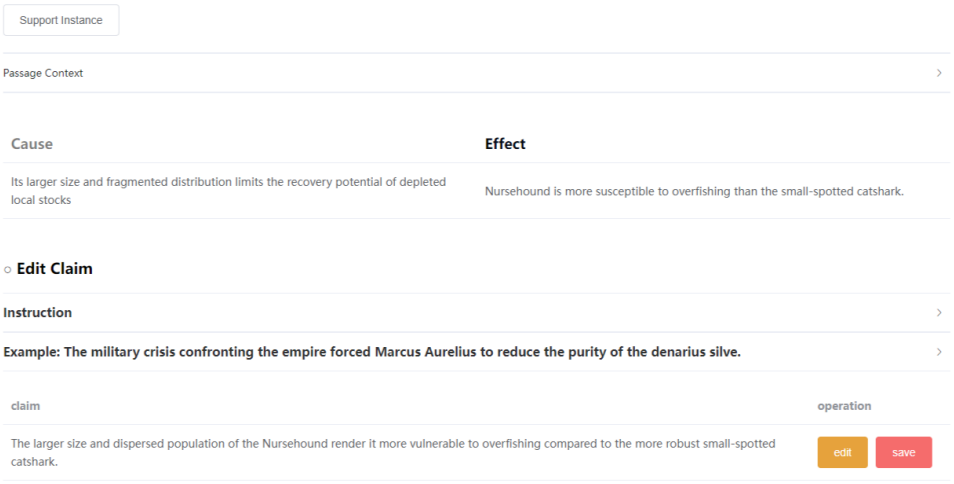

Figure 8: Claim Annotation Interface

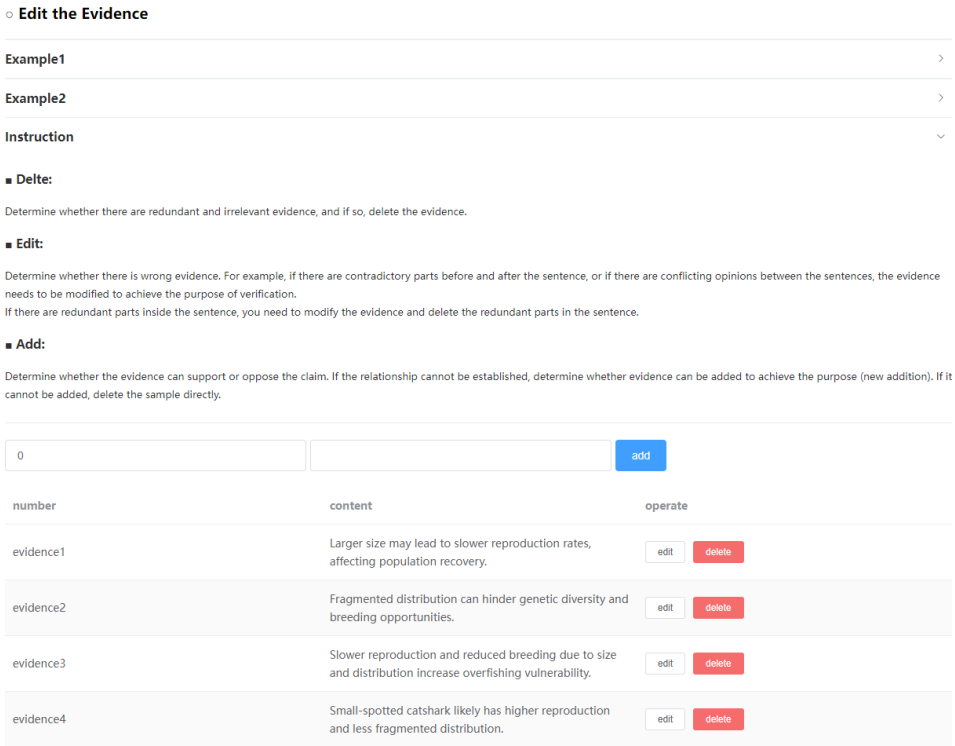

Figure 9: Evidence Annotation Interface

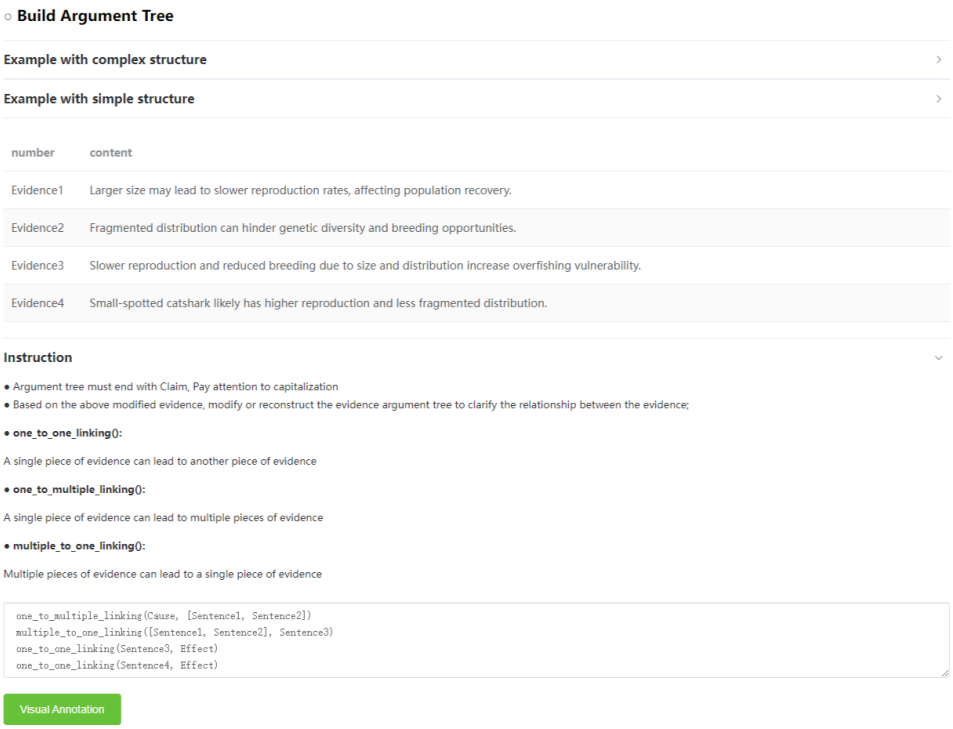

Figure 10: Argument Structure Annotation Interface

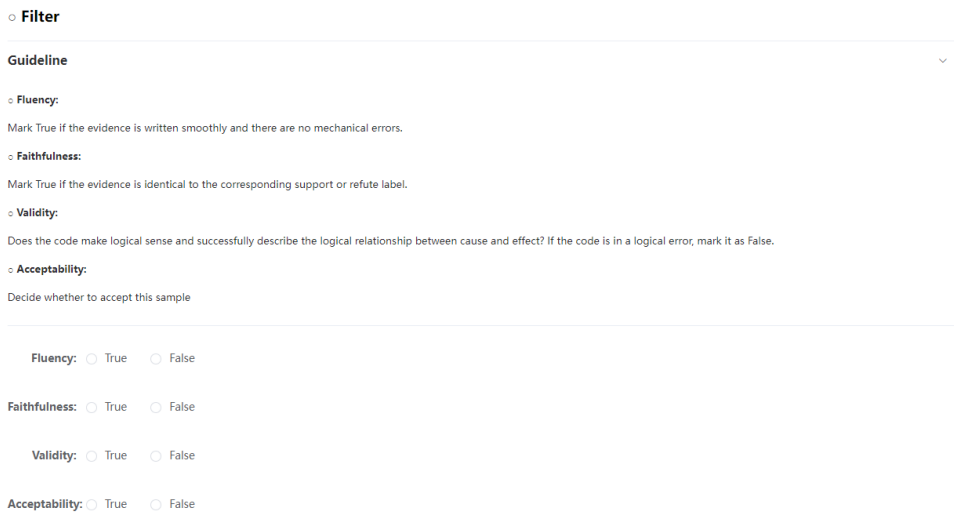

Figure 11: Filter Annotation Interface

I am an excellent linguist. Your task is to generate a refined claim by synthesizing the content of the {Cause} and the {Effect}.
Be sure to use creative and diverse ways to generate the {Claim}.
You will be provided with
{**Cause**} is the reason why the {Effect} is established;
{**Effect**} is the consequence of {Cause}.
Note:
No matter whether this pair of {Cause} and {effect} is correct or not, you must synthesize a concise and clean {Claim} with causality contained.
Do not induce any extra information in your generation except what we provided.
Example:
Cause: Powerranger's film market is in the doldrums.
Effect: It's not possible that the Power Rangers series will get a sequel.
Claim: The lackluster performance of the Power Rangers film in the market has plunged its sequel prospects into uncertainty.
...

Table 11: Prompt for Claim Generation

I am an excellent logician. Your task is to generate persuasive factual evidence by expanding the {anchor explanation} to explain why a {cause} can lead to the {effect} in a step-by-step manner. The {anchor explanation} provides the basic explanation you should refer to and generate around. Solving this task with multiple steps, each step includes interleaving {Thought}, {Action}, {Inference}.
{**Thought**} is the analysis of the previous step by combining the information of {cause} {effect} pair, the generated entailment {Inference} in previous steps, and the {anchor explanation};
{**Thought**} is also to infer which {Action} should be taken next to reach the {effect} from the current step;
{**Inference**} is the result of the current step when taking the {Action};
{**Action**} is the solution you can operate based on the {Thought} in current step, and consists of 3 types:
(1) {**Further Reasoning**}: infer directly using the content of the previous {Inference}, {cause}, and the {anchor explanation}.
(2) {**Add new condition**}: provide new factual condition to support the {cause} {efect} pair. Do this step only when you need context information that has not been provided in the {cause} {effect} pair and the {anchor explanation}, to make the inference logically and reasonably.
(3) {**Finish**}: summary the final results based on the {Inference}.

(Don't simply list facts, but reason in a detailed and step-by-step manner.)

Example:
cause: Virtue of compassion for all living things in Buddhism.
effect: Qisong argued that Buddhist ethics were superior to Confucian ethics.
anchor explanation: Confucian ethics do not dictate compassion for all living things.
Building support reasoning process:
Thought1: By referring to the anchor explanation and the Cause, the difference between Buddhism and Confucianism towards compassion for all living things leads to the Qisong arguing that Buddhist ethics are superior to Confucian ethics. To support the cause-effect pair, I need to understand in detail what would make Qisong argue Buddhism against Confucian ethics.
Action1: Further Reasoning
...
Thought5: Based on the above reasoning, I can conclude that the final effect is established: Qisong argued that Buddhist ethics were superior to Confucian ethics.
Action5: Finish
Summary: Based on the Inference1, Inference2, Inference3, and Inference4, the cause-effect pair is supported.

(Every Inference has a maximum of 25 words)
(The reasoning process should be completed in multiple steps. The information in each step should be as concise as possible, and no redundant information should be included between each step.)
...

Table 12: Prompt for Evidence Generation

I am an excellent linguist. Your task is to generate an opposite sentence.
Given an original sentence, you should generate a new sentence to refute this sentence.
Be sure to use creative methods instead of simply adding negative words.
You will be provided with
{**Original sentence**} is the original sentence we provide;
{**Opposite sentence**} is the Opposite sentence, which will refute the original sentence.
Mandatory constraints: no unnecessary extra information can be induced.

Example:
Original sentence: It was unlikely that any sequels would be made in the Power Rangers series.
Opposite sentence: It's still possible that the Power Rangers series will get a sequel.
...
#Just put the output after {Opposite sentence}. you are not allowed to do anything else.

Table 13: Prompt for Counterfactual Effect Generation

class FACT
<FACT CLASS PROMPT>
I am an excellent logician and programmer.
<TASK PROMPT>
Example:
Claim = FACT("...")
Sentence1 = FACT ("...")
Sentence2 = FACT ("...")
Sentence3 = FACT ("...")
Sentence4 = FACT ("...")
def build_tree_proof():
    one_to_multiple_linking(Sentence2, [Sentence3, Sentence4])
    multiple_to_one_linking([Sentence1, Sentence3, Sentence4], Claim)
...

Table 14: Argument Structure Generation Prompt

I am an excellent logician and fact checker.
My task is to verify whether the claim is {SUPPORTS}, {REFUTES} or {NOT ENOUGH INFO}, based on {Evidence}.
No need to focus on whether the given evidence is correct.
(No need to return anything else superfluous)
Example:
 # Input:
Claim: Joanna Briscoe attributes the novel 'The Great Lover's evocative "sense of time and place" to its adept treatment of distinct elements such as Fabianism and class politics in that British era.
 # Evidence:
...
Determine whether {Claim} is supported or refuted:
 # Output:
{SUPPORTS}

Table 15: Prompt for Task1

<FACT CLASS PROMPT>

<TASK PROMPT>

Example:
 # Input:
Claim = FACT("...")
 # Evidence:
Sentence1 = FACT("...")
...

def build_tree_proof():
    one_to_one_linking(Sentence13, Sentence16)
    one_to_one_linking(Sentence13, Sentence17)
    one_to_one_linking(Sentence13, Sentence18)
    one_to_one_linking(Sentence11, Sentence17)
    multiple_to_one_linking([Sentence16, Sentence17, Sentence18], Claim)

Determine whether {Claim} is supported or refuted:
 # Output:
print("REFUTES")

Table 16: Prompt for Task2

```
class FACT:
    """
    self.fact is the factual sentence.
    self.children is the child node that is to be linked.
    self.parents is the parent node that links the children node.
    """
    def __init__(self, fact: str):
        self.fact = fact
        self.children = []
        self.parents = []

def multiple_to_one_linking(p_fact_list: List[FACT], c_fact: FACT):
    """
    Implement the logic to link multiple parent nodes to a single child node.
    p_fact_list contains the list of parent nodes.
    c_fact is the child node.
    """
    assert len(p_fact_list) > 1
    for p_fact in p_fact_list:
        p_fact.children.append(c_fact)
        c_fact.parents.append(p_fact)

def one_to_one_linking(p_fact: FACT, c_fact: FACT):
    """
    Implement the logic to link a single child node to a single parent node.
    p_fact is the parent node.
    c_fact is the child node.
    """
    p_fact.children.append(c_fact)
    c_fact.parents.append(p_fact)

def one_to_multiple_linking(p_fact: FACT, c_fact_list: List[FACT]):
    """
    Implement the logic to link a single parent to multiple children.
    p_fact is the parent node.
    c_fact_list contains the list of children nodes.
    """
    assert len(c_fact_list) > 1
    for c_fact in c_fact_list:
        c_fact.parents.append(p_fact)
        p_fact.children.append(c_fact)
```

Figure 12: Argument Structure Generation Fact Class Prompt

I am an excellent logician and programmer.
The input is an instance of {CAUSE}, an instance of {EFFECT} and an explanation consist of multiple {FACT} instances.
Your task is to complete a {build_tree_proof()} function by reasoning the relationship among the {CAUSE} instance, {EFFECT} instance, and multiple {FACT} instances, and linking these instances to form a Proof Structure logically and formally.
Building the Proof Structure by calling {multiple_to_one_linking}, {one_to_one_linking}, and {one_to_multiple_linking}, which aims to describe the reasoning process that why the {CAUSE} leads to the {EFFECT} by using the intermediate {FACT} in a step-by-step manner.
Only when you are sure that the parent node can infer the child node, you can connect them.
{multiple_to_one_linking} is to implement the logic to link multiple parents to a single child.
{one_to_one_linking} is to implement the logic to link a single child to a single parent.
{one_to_multiple_linking} is to implement the logic to link a single parent to multiple children.

Consider the following aspects of your generation, this is very important to me:
1. The first line of {build_tree_proof()} does not need to start with {CAUSE}, but must end with {EFFECT}.
2. {CAUSE} node and {EFFECT} node must be used, ensuring {EFFECT} only be used ones in the Proof Structure.
3. If a parent node can infer multiple children nodes, be sure to use {one_to_multiple_linking}, instead of {one_to_one_linking}.

Figure 13: Argument Structure Generation Task Prompt Example

<FACT CLASS PROMPT>

<TASK PROMPT>

Example:
 # Input:
Claim = FACT("...")
 # Evidence:
Sentence1 = FACT("...")
...
# Determine whether {Claim} is supported or refuted, complete this function:
def build_tree_proof():
 # Output:
```python
one_to_one_linking(Sentence8, Sentence15)
one_to_one_linking(Sentence15, Sentence16)
one_to_one_linking(Sentence16, Sentence17)
one_to_one_linking(Sentence17, Sentence18)
one_to_one_linking(Sentence18, Claim)
print("SUPPORTS")
```

Table 17: Prompt for Task3

<FACT CLASS PROMPT>

<TASK PROMPT>

Example:
 # Input:
Claim = FACT("...")
 # Evidence:
Sentence1 = FACT("...")
...
# Determine whether {Claim} is supported or refuted, complete this function:
def build_tree_proof():
 # Output:
```python
Sentence15 = FACT("...")
one_to_one_linking(Sentence8, Sentence15)
Sentence16 = FACT("...")
one_to_one_linking(Sentence15, Sentence16)
Sentence17 = FACT("...")
one_to_one_linking(Sentence16, Sentence17)
Sentence18 = FACT("...")
one_to_one_linking(Sentence17, Sentence18)
one_to_one_linking(Sentence18, Claim)
print("SUPPORTS")
```

Table 18: Prompt for Task4