Towards Generalizable and Faithful Logic Reasoning over Natural Language via Resolution Refutation

Zhouhao Sun[1], Xiao Ding[1]*, Li Du[2], Bibo Cai[1], Jinglong Gao[1], Ting Liu[1], Qin Bing[1]
[1]Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
[2]Beijing Academy of Artificial Intelligence, Beijing, China
{zhsun, xding, bbcai, jlgao, tliu, bqin}@ir.hit.edu.cn
duli@baai.ac.cn

Abstract

Large language models (LLMs) have achieved significant performance in various natural language reasoning tasks. However, they still struggle with performing first-order logic reasoning over formal logical theories expressed in natural language. This is because the previous LLMs-based reasoning systems have the theoretical incompleteness issue. As a result, it can only address a limited set of simple reasoning problems, which significantly decreases their generalization ability. To address this issue, we propose a novel framework, named Generalizable and Faithful Reasoner (GFaiR), which introduces the paradigm of resolution refutation. Resolution refutation has the capability to solve all first-order logic reasoning problems by extending reasoning rules and employing the principle of proof by contradiction, so our system's completeness can be improved by introducing resolution refutation. Experimental results demonstrate that our system outperforms previous works by achieving state-of-the-art performances in complex scenarios while maintaining performances in simple scenarios. Besides, we observe that GFaiR is faithful to its reasoning process.

Keywords: logic reasoning, resolution refutation, completeness, faithfulness, large language models

1. Introduction

The natural language-based logical reasoning task requires the model to understand the abstract logical relationships within statements expressed in natural language to deduce a conclusion. For example, as shown in Figure 1a, the task is to determine the value of the hypothesis (True, False, Unknown) based on a natural language theory (**NL Theory**) which consists of a set of rules and facts explicitly stated in natural language. This task is increasingly gaining attention (Sun et al., 2021; Kazemi et al., 2023), as it bridges the natural language with abstract logical thinking, which plays a pivotal role in complex problem-solving and cognitive reasoning.

Recently, transformer-based LLMs have achieved significant performance in various natural language reasoning tasks (Wei et al., 2022; Qiao et al., 2022). Theoretical analyses have also demonstrated that transformers have the potential to perform logical reasoning over formal theories (Schlegel et al., 2022; Zhang et al., 2023). However, it still remains challenging for the present LLMs (Pan et al., 2023; Kazemi et al., 2023), even for the State-of-the-Art models including ChatGPT (Bang et al., 2023). This is because of the *hallucination* problem (Golovneva et al., 2023; Ribeiro et al., 2023), i.e., LLMs may hallucinate incorrect intermediate reasoning steps to draw the final conclusions. As a result, the inference
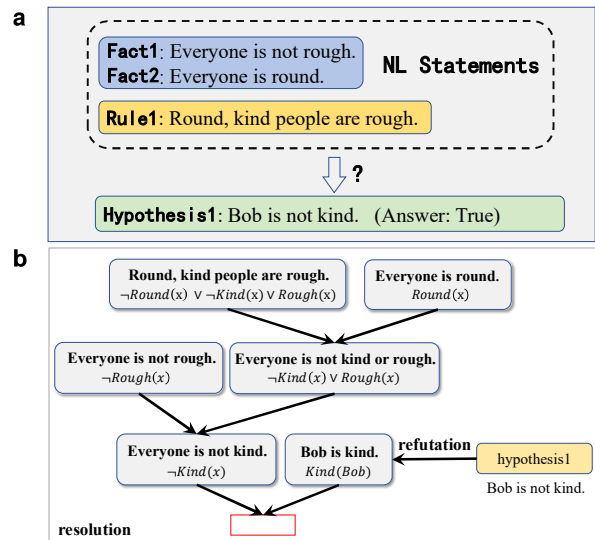
*Corresponding Author

Figure 1: (a) Example of an NL Theory and a hypothesis with gold answers. Note that the meaning of these statements are not related to the common sense. (b) For hypothesis 1, the reasoning process using the method of resolution refutation is shown. The process of refutation is reflected from 'hypothesis' to "Bob is kind" and the grey box represents the process of resolution at the natural language level.

results are not *faithful* to be trusted (Lyu et al., 2023; Creswell and Shanahan, 2022). Moreover, if regarding large language models as inference sys-

tems, hallucination will affect their *completeness*. A complete inference system means that **all** the hypotheses with determined labels can be inferred by applying valid reasoning rules contained in the inference system. However, the hallucination problem prevents the LLMs from correctly wielding reasoning rules to draw conclusions, thus leading to incompleteness inference systems.

To reduce hallucination and improve faithfulness for LLMs, previous works mainly enhance the reasoning process of LLMs by a stepwise inference paradigm. According to the direction of reasoning, these works can be divided into two groups. The forward chaining approach (Sanyal et al., 2022) starts from known rules to check if there exists any rule whose conditions are all satisfied by the given facts, if so, we apply the reasoning rule of forward chaining to derive a new conclusion, this procedure continues until no new conclusions can be drawn or the hypothesis is proved. The backward chaining approach (Qu et al., 2022) starts from the hypothesis and reasons in an opposite direction to derive a set of facts that need to be satisfied, then querying if these inferred facts overlap the known facts. By introducing intermediate steps, faithfulness can be improved.

However, the performance of these methods in complex logical reasoning scenarios is still unsatisfying. In some cases, their performance may be lower than using LLMs alone, or even lower than random guesses. This is caused by the inherent deficiency of these methods that the forward or backward reasoning method is **incomplete**. It means that there will be some hypotheses with determined values that are considered Unknown by the model. As a result, it can only accommodate relatively simple scenarios. Take forward chaining as an example, forward chaining is incomplete because it is capable of reasoning if and only if 'all the conditions of a certain rule can be proven to be true based on known facts' (condition 1). However, In the process of reasoning, there are some exceptional cases where forward chaining cannot reason. For the Hypothesis 1 in Figure 1a, forward chaining is unable to complete this type of reasoning since the condition of the rule "kind people" cannot be proven to be true by the facts. Hence, no conclusions can be drawn and hypothesis 1 will be considered Unknown. For the backward chaining, inference also cannot be made since hypothesis 1 "not kind" does not appear at the right hand of the rule. Hence the hypothesis will also be considered Unknown.

Inspired by the logical reasoning methods in the field of symbolic logic, we attempt to introduce a complete logical reasoning paradigm (under first-order logic) resolution refutation (Russell, 2010) whose reasoning procedure is not constrained by

the condition 1 to improve completeness, and propose a novel reasoning framework GFaiR. Figure 1b illustrates the reasoning process of our model. For hypothesis 1, by utilizing the reasoning rule of resolution, we can derive 'Everyone is not kind' step by step from the known information by performing resolution at the natural language level. Then by refutation, 'kind' appears in the known information so we can finally prove that hypothesis 1 is True. As a result, the combining of resolution refutation enables the model to handle more complex reasoning scenarios and enhances its generalization ability. Because the process of resolution refutation is complex, so we detail them in Section 2.

To combine resolution, we need to first select two theories and then utilize a reasoning model to perform resolution over them at the natural language level. However, the previous (Sanyal et al., 2022) transformers-based selection module only considers selecting which theories are more likely to infer the target hypothesis, without taking into account whether these two theories are logically related. This leads to scenarios where the selected theories are completely unrelated, which further causes the failure of resolution and the generation of invalid conclusions which may result in hallucinations. As a result, we use a validity contrastive loss-based verifier to distinguish valid conditions from illogical statements. This ensures that a valid conclusion can be drawn from the selected theories through logical reasoning, thereby providing guarantees for resolution and improving faithfulness by reducing hallucinations.

We validate our method on the widely adopted Ruletaker dataset and a more challenging Hard Ruletaker dataset, as well as the natural language satisfiability (NLSAT). Experimental results show that our approach is faithful to its reasoning process and has stronger in-domain inference accuracy, meanwhile demonstrating zero-shot generalization performances[1].

2. Background

Natural Language Reasoning with First-Order Logic We follow the task definition proposed by Han et al. (2022). Given a hypothesis H and an NL Theory NLT (including a series of facts and rules expressed in natural language) without contradiction, the goal is to determine the value of H: True, False, or Unknown. Note that NLT and H are annotated with parallel FOL (first-order logic) Theory and FOL hypothesis, and the value is determined by the FOL reasoning result of the FOL Theory and FOL hypothesis. If the value is True or False, it is

---

[1]The source code of GFaiR has been made available at https://github.com/spirit-moon-fly/GFaiR.

expected to give a reasoning process, which consists of a series of reasoning steps $(p_1, p_2, ..., p_n)$, and each reasoning step p_i includes selected rules or facts s_i along with reasoning conclusion c_i.

Resolution Refutation Resolution refutation (Nawaz et al., 2019) is a commonly used and complete reasoning method under first-order logic, i.e. for a hypothesis whose label is True or False under the semantics of full FOL, applying the reasoning method of resolution refutation can infer the label of the hypothesis. Let F be the FOL formula set of the given premises, and Q be the hypothesis, then the process of proving that Q is True by resolution refutation is shown as follows:

1) Negate Q to get $\neg Q$, and merge it into the formula set F to get $\{F, \neg Q\}$.
2) Transform $\{F, \neg Q\}$ into a clause set in Skolem normal form. (Skolem standardization)
3) Apply resolution principle (Robinson, 1965) to resolve clauses in the clause set, each resolution step generates a resolved clause, which is then added to the clause set. This process is repeated iteratively. If an empty clause is obtained during the resolution step, it indicates a contradiction in the clause set and proves that Q is True.

The process of proving that Q is False is similar. Therefore, when dealing with our target task, we can determine the value of H by applying a reasoning model to the theory set T_1 composed of NLT and H and the theory set T_2 composed of NLT and $\neg H$ at the same time, where the reasoning model implicitly performs resolution at the natural language level. If there is no contradiction in the theory set T_1 and there is a contradiction in the theory set T_2, it proves that H is True. On the contrary, it proves that H is False. if there are no contradictions in two theory sets, H is Unknown. an example of resolution refutation reasoning procedure can be seen in Section 3.3.

3. Method

3.1. Overview

Although improved faithfulness compared with vanilla LLMs, existing stepwise inference methods based on forward or backward chaining are incomplete, which makes them unable to generalize to complex reasoning scenarios.

In this paper, we propose a novel reasoning framework GFaiR. As shown in Figure 2, GFaiR introduces resolution refutation to improve the completeness.

Specifically, GFaiR is composed of five modules: (1) A converter for augmenting the given NL Theory with the negated hypothesis and to convert the representations of natural language for resolution at the natural language level in the following

reasoning process. (2) A pre-selector to select a theory for drawing intermediate conclusions. (3) A post-selector to select another theory by explicitly modeling the relationship between the theory selected by the pre-selector and the remaining ones. (4) A knowledge composer to generate a novel conclusion by applying the resolution rule at the natural language level. (5) A verifier to ensure that a valid conclusion can be drawn from the selected theories through logical reasoning, thereby providing guarantees for resolution and improving faithfulness.

In the following sections, we will first introduce the architecture of GFaiR, and then explain the inference and training procedure of GFaiR.

3.2. Architecture

Overall, GFaiR is an iterative model where the one-hop intermediate conclusions are generated step-by-step. Our model is shown in Figure 2. Specifically, we have the following five modules:

Converter Given the NL Theory and hypothesis, before directly performing reasoning, we first employ a T5-based converter to automatically convert the hypothesis into its negation form for refutation in the inference process (not reflected in Figure 2). Additionally, because our knowledge composer mimics the resolution step which cannot deal with existential quantifiers and some implicit logical relationships such as $\rightarrow$, we need a step to convert the implicit logical relationships and existential quantifiers while retaining as much of the original text as possible. The convertor can also perform this step by imitating the Skolem standardization step in resolution refutation, which transforms NL Theory and hypothesis (or its negation form) into natural language representations similar to the Skolem normal form. As shown in Figure 2, the converter converts 'Round, kind people are rough' to 'Everyone is not kind or not round or rough'. The converted NL Theory and hypothesis will be taken as inputs for the following reasoning process.

Pre-Selector (Pre-S) The pre-selector is an XLNET-based (Yang et al., 2019) classification model that takes the concatenated theories in the theory set as input (including intermediate conclusions, converted NL Theory and hypothesis), and selects a theory for generating new conclusions in the current iterative steps. Taking theory set $T = \{t_1, t_2, ..., t_n\}$ in Figure 2 as an example, we concatenate and separate them with the [SEP] token to form the input $[CLS] [t_i [SEP]]_n$ ($[\ ]_n$ denotes continued concatenation). The output is a one-dimensional vector denoted as u, which is obtained by classifying each [SEP] token embedding via a linear binary classification layer. During iteration, we select the theory in front of the [SEP] token corresponding to the maximum value in the vector u. The example in Figure 2 illustrates the selection
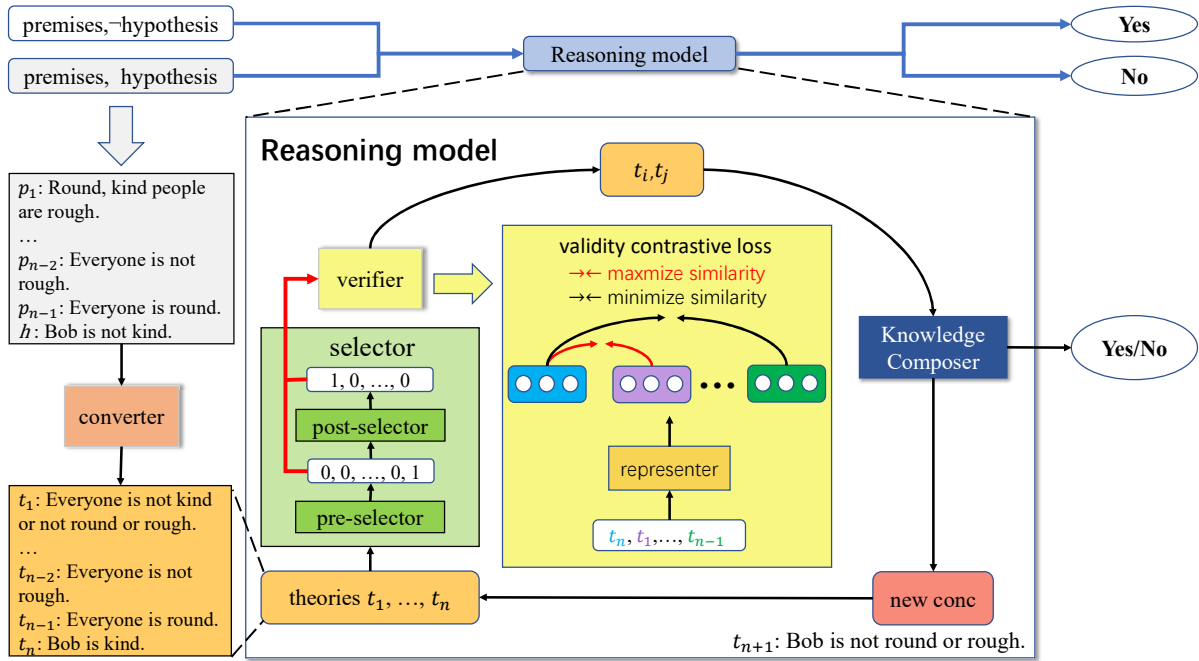
16529

Figure 2: Architecture of GFaiR. We mark the converter in orange, the selector (consisting of pre-selector and post-selector) in green, the verifier with its validity contrastive loss in yellow, and the knowledge composer in blue, respectively.

of t_n based on the highest value in u.

Post-Selector (Post-S) The post-selector is also an XLNET-based classification model aiming to select another theory based on the theory selected by the pre-selector and the remaining theories. We designed this module to explicitly model the relationship between the theory selected by the pre-selector and the remaining ones. As shown in Figure 2, t_n is the theory selected in the previous step, and then place t_n at the beginning of the input, while keeping the order of the other theories unchanged and concatenating them after t_n. We also use [SEP] token to separate these theories to form the input $[CLS]\ t_n\ [SEP]\ [t_i\ [SEP]]_{n-1}$. The output is a one-dimensional vector v, which is obtained by classifying each [SEP] token embedding (except the first [SEP] token) via a linear binary classification layer. Similar to the pre-selector, the example in Figure 2 illustrates the selection of t_1 according to the value in vector v.

Knowledge Composer (KC) The knowledge composer is a generative transformer T5 that can learn the resolution rule implicitly from data, and apply the learned resolution rule at the natural language level to generate a novel conclusion. As shown in Figure 2, the input is two theories selected by the pre-selector and post-selector (t_n and t_1), and the output t_{n+1} is an intermediate conclusion expressed in natural language, which will be merged into the theory set.

Verifier The previous (Sanyal et al., 2022)

transformers-based selection module is not accurate enough for resolution refutation, which leads to scenarios where the selected theories are unrelated. This causes the failure of resolution and further the generation of invalid conclusions which may result in hallucinations. As a result, we use a validity contrastive loss-based verifier to verify two theories selected by the pre-selector and post-selector to ensure that a valid conclusion can be drawn from them through logical reasoning, thus providing guarantees for resolution and improving faithfulness by reducing hallucinations. The validity contrastive loss is shown in Figure 2:

To facilitate explanation, we establish the following definitions: A **theory pair** (t_i, t_j) composed of two theories t_i and t_j is **valid** if and only if a valid conclusion can be drawn from them through logical reasoning. Because our knowledge composer emulates the resolution step at the natural language level to draw intermediate conclusions, the criterion for determining if the theory pair is valid lies in whether the FOL expressions corresponding to these two theories can be used for resolution.

We consider all the theory pairs composed of the theory selected by pre-selector and the remaining ones, and then devise validity contrastive loss by maximizing the cosine similarity of valid theory pairs (pink and blue in Figure 2) while minimizing the cosine similarity of invalid theory pairs (green and blue in the Figure 2). Please refer to 3.4 for the loss function.

When verifying the theory selected by the post-selector t_k, the verifier first calculates the cosine similarity between t_k and the theory selected by the pre-selector t_m. If the similarity score is above 0 (similarity score is within the range of -1 to 1), the theory pair is deemed valid and selected as input for the knowledge composer. Conversely, it is invalid and the post-selector will select a new theory for verification. This process continues until a theory is selected that can form a valid theory pair with t_m.

3.3. Inference

During inference, the converter first converts the NL Theory and hypothesis into two theory sets represented in natural language similar to the skolem normal form. One of them consists of the NL Theory and hypothesis, the other consists of the NL Theory and the negation of the hypothesis. Then we apply our reasoning model (shown in Figure 2) to two theory sets separately to infer if there exists a contradiction, which determines the value of the hypothesis (referring to the background for more details). Since our model is a neural network model rather than a symbolic reasoning system, there are accidental conditions that contradiction exists in both theory sets. In such cases, we employ a heuristic approach to determine the value of the hypothesis (according to the number of reasoning steps). Below we will explain how to infer a contradiction in a theory set.

For a specific theory set T, the pre-selector first selects a theory t_i. Then, under the guidance of the verifier, the post-selector selects a theory t_j that can form a valid theory pair with t_i. If it does not exist, stop and conclude that there are no contradictions in the theory set. Conversely, the knowledge composer composes two selected theories to generate a new conclusion. If the conclusion is an empty string (corresponding to the empty clause in the process of resolution refutation), it indicates that there is a contradiction in the theory set and stops the iteration. Otherwise, the newly generated conclusion is placed in the theory set T to participate in the following reasoning process. For the example in Figure 2, the reasoning model will first derive $'t_{n+1}$: Bob is not round or rough.' by resolving $'t_1$: Everyone is not kind or not round or rough.' and $'t_n$: Bob is kind.'. Then, by combining t_{n+1} and $'t_{n-2}$ Everyone is not rough.', the model can derive $'t_{n+2}$: Bob is not round.'. Finially, we can derive an empty string from t_{n+2} and $'t_{n-1}$: Everyone is round.', which indicates a contradiction in the theory set and illustrated that the hypothesis is True. Due to the infinite search space of first-order logical reasoning, we design a maximum number of reasoning steps N. When it is reached, we assume that there are no contradictions in the theory set

and stop iteration. There may be cases where no theories are selected to form a valid theory pair. For example, the theory set is: {Bob is kind. Bob is tall. Bob is happy.}. In this situation, we are unable to derive a valid conclusion based on any theory pair. Therefore, we cannot derive any valid conclusions, and we will halt the search and consider that there are no contradictions in this theory set.

3.4. Training

Each component of our model is trained separately. The training data of the converter is every fact and rule in the NL Theory and hypothesis as well as their corresponding natural language representations similar to the skolem normal form (or the negation). The following mainly introduces the training methods of the other four modules.

From Background, we know that the resolution refutation process for proving a hypothesis is True or False involves proving a theory set is contradictory. And each step for proving a theory set $T = \{t_1, ..., t_n\}$ is contradictory can be represented as (t_i, t_j, t_k), which means that the intermediate conclusion t_k is generated based on t_i and t_j already existed in the theory set T (intermediate conclusions generated by previous reasoning steps have been merged into the theory set T). Then, for a theory set T with contradiction and one of its reasoning steps $\{t_i, t_j, t_k\}$, we can generate four training samples for training Pre-Selector, Post-Selector, and Knowledge Composer, respectively:

$$Pre\text{-}S\ Input = \{T\}\,;Pre\text{-}S\ Output = \{t_i, t_j\}$$
$$Post\text{-}S\ Input = \{T, t_i\}\,;Post\text{-}S\ Output = \{t_j\}$$
$$Post\text{-}S\ Input = \{T, t_j\}\,;Post\text{-}S\ Output = \{t_i\}$$
$$KC\ Input = \{t_i, t_j\}\,;KC\ Output = \{t_k\}$$

The generative knowledge composer can learn the resolution rule implicitly after training by language modeling loss. The pre-selector and post-selector are classification models, so that their output is converted to class labels instead of text. We use binary cross entropy loss to train these two modules.

To train the verifier, we utilize the output of XL-NET in post-selector, specifically the vector representation corresponding to the [SEP] token via a linear layer, as the vector representations of the theories for simplicity. So the verifier and post-selector are trained jointly, with their loss function combined using a hyperparameter α. For the example in Figure 2, $T = \{t_1, ..., t_n\}$ is the current theory set, $V = \{v_1, ..., v_n\}$ is the corresponding vector representations, t_n is the theory selected by the pre-selector. Assume that $P = \{p_1, ..., p_k\}$ represents the indices of theories that can form a valid theory pair with t_n, which are considered as positive examples, $R = \{r_1, ..., r_m\}$ represents the indices of theories that cannot, which are considered as

negative examples. The specific definition of the validity contrastive loss (VCE) is shown as follows, where the maximum similarity between positive examples is constrained to be 0.8 to prevent model collapse:

$$L_{vce} = -\frac{1}{k}\sum_{j=1}^{k} log \frac{exp(max(sim(v_n, v_{p_j}), 0.8))}{\sum_{i=1}^{m} exp(sim(v_n, v_{r_i}))}$$

4. Experiment Setup

Tasks and Datasets Following Richardson and Sabharwal (2022), we trained and evaluated on the easy Ruletaker-depth-3ext dataset (Tafjord et al., 2021), then tested on the test set of Ruletaker-depth-3ext and Ruletaker-depth-5 dataset (later we will refer to them as Ruletaker-3ext and Ruletaker-D5) as well as the dev set of Hard Ruletaker (Hard RuleTaker only have the dev set). **Hard Ruletaker** is a **harder** dataset by eliminating potential bias (Morishita et al., 2023) compared to Ruletaker-3ext and Ruletaker-D5. However, the Hard Ruletaker dataset only includes True and False labels without Unknown labels, which may not accurately reflect the ability of the model. So we use the same method to sample hard instances whose label is Unknown and added it to Hard Ruletaker dataset to balance three kinds of labels. We named the new dataset **Hard Ruletaker***. To further evaluate the performance of our model after training on hard instances, we divide Hard Ruletaker* to train, dev, and test set based on a ratio of 8.5,0.5,1, the divided dataset was called **Hard Ruletaker****. However, these datasets do not contain existence quantifiers that are implicitly expressed in natural language, so we also use a method similar to Richardson and Sabharwal (2022) to construct a dataset with existence quantifiers called **Ruletaker-E**.

To train GFaiR, we first get each data's FOL representation and then employ a resolution refutation based FOL prover to automatically derive the intermediate reasoning process. Finally, we transform it from FOL representations into natural language representations by using natural language templates. More details can be seen in Appendix A.

Baselines For the first task, we compare GFaiR with two kinds of methods:

(1) **Pretrained Language Model Based Methods:** We use Roberta-large (Liu et al., 2019), T5-large and ChatGPT (gpt-3.5-turbo) as baselines. For Roberta-large and T5-large, we finetune them on the Ruletaker-3ext and Hard Ruletaker** datasets. For ChatGPT, we use the method of instruct and chain-of-thought prompt to evaluate its performance. Due to cost reasons, we respectively tested 3000 pieces of data in three datasets.

(2) **Stepwise Inference Methods**: we mainly compare GFaiR with the model combined with forward chaining FaiRR (Sanyal et al., 2022) and the

Model	Ruletaker-3ext		Hard RT		Hard RT*	
	EA	FA	EA	FA	EA	FA
T5	97.7	—	57.3	—	57.5	—
Roberta	98.9	—	59.6	—	59.7	—
ChatGPT	56.5	42.8	57.0	2.7	38.9	6.9
IBR	98.9	98.1	59.6	12.1	59.7	29.6
FaiRR	99.0	98.4	14.1	12.2	41.1	39.8
NLProofs	**99.3**	**99.2**	14.3	13.8	41.8	41.4
GFaiR	98.1	98.0	**68.5**	**67.5**	**73.9**	**71.7**

Table 1: Comparison of GFaiR with baselines when trained on Ruletaker-3ext and tested on Ruletaker-3ext and two hard datasets. EA, FA, Hard RT, and Hard RT* refer to entailment accuracy, full accuracy, Hard Ruletaker, and Hard Ruletaker* respectively.

model combined with backward chaining IBR (Qu et al., 2022). And we also compare GFaiR with NLProofs (Yang et al., 2022) which conducts proof search on partial proof graphs. More details can be seen in Appendix B.

Evaluation protocol Following Qu et al. (2022), We consider two main aspects for evaluating the model's performance in our study: (1) **Entailment accuracy (EA)** measures how accurately the model is able to predict the label of the hypothesis. (2) **Full accuracy (FA)** measures how accurately the model can simultaneously predict the label and the valid proof (i.e. the reasoning process) of the hypothesis. For a reasoning process $P = (p_1, p_2 \ldots p_n)$, it is valid if and only if every reasoning step p_i is correct. A reasoning step p_i includes selected rules or facts s_i, along with reasoning conclusion c_i. To check if p_i is right, we use the FOL format expression of s_i and c_i, denoting as fs_i and fc_i. And we consider p_i is right if fc_i can be directly derived by fs_i using a valid reasoning rule under FOL. Following Tafjord et al. (2021), when the model predicts Unknown, no proof will be generated and we think the proof is right when the gold label is Unknown. Note that our method for evaluating the reasoning process is more flexible compared to the evaluation method proposed in previous work (Saha et al., 2020), which relies on precisely matching between the gold proof and the predicted proof. Instead, our evaluation method is able to take different reasoning paths into account. However, our method still will not evaluate incorrect reasoning processes as correct ones ensured by symbolized logical reasoning.

5. Experiment Results

5.1. Main results

To investigate different methods' in-domain performance on easy problems and zero-shot generaliza-

depth	FaiRR		NLProofs		GFaiR	
	EA	FA	EA	FA	EA	FA
N/A	99.4	99.4	99.4	99.4	96.2	96.2
0	100	100	100	100	99.9	99.9
1	99.5	99.2	99.9	99.9	99.5	99.5
2	98.4	96.0	99.0	99.0	98.2	97.9
3	93.1	84.8	94.1	93.4	95.8	95.1
4	88.8	77.3	79.5	77.2	94.2	92.5
5	78.7	67.8	69.6	57.3	94.2	91.9

Table 2: Depth-wise results on Ruletaker-D5, N/A represents the depth is unknown because the value of the hypothesis is 'unknown'.

Model	Hard RuleTaker**		RuleTaker-E	
	EA	FA	EA	FA
T5	87.1	—	75.7	—
Roberta	89.3	—	76.8	—
IBR	89.3	39.2	76.8	35.3
FaiRR	40.4	34.0	38.4	36.6
NLProofs	40.7	39.4	38.6	38.2
GFaiR	**92.2**	**92.2**	**83.2**	**82.7**

Table 3: Results on Hard Ruletaker** and Ruletaker-E dataset.

tion ability on hard problems, we trained and evaluated on easy Ruletaker-3ext dataset, and then tested on Ruletaker-3ext and two hard datasets (Hard Ruletaker and Hard Ruletaker*). Results are shown in Table 1, from which we can observe that:

(1) Compared to pretrained language model based methods (T5-large, Roberta-large, and Chat-GPT), we can find that stepwise inference methods are more faithful than ChatGPT from the difference between the value of EA and FA.

(2) Compared to stepwise inference methods IBR, FaiRR, and NLProofs, GFaiR shows comparable performance on the biased RuleTaker-3ext dataset, and significantly outperforms on two debiased hard datasets, which demonstrates stronger zero-shot generalization performance according to EA and FA. This suggests that by introducing resolution refutation to improve completeness, the stepwise inference methods can generalize to complex logical reasoning scenarios. In contrast, previous stepwise inference methods IBR, FaiRR, and NLProofs are incomplete and often classify hypotheses that can be inferred as True or False as Unknown, so they exhibit unsatisfied when dealing with complex logical reasoning scenarios. Hence GFaiR's ability to generalize to complex logical reasoning scenarios is better.

(3) From the difference between the value of EA and FA, we can observe that our model is faithful. Though this difference of FaiRR as well as GFaiR on two hard datasets is much smaller, their entailment accuracy is relatively low so there is no point in only considering their faithfulness. However, GFaiR both achieves higher entailment accuracy and maintains faithfulness by combining resolution refutation and introducing a validity contrastive loss-based verifier.

(4) ChatGPT does not outperform other models significantly and even performs worse than them. On the one hand, this reflects the difficulty of this task. On the other hand, this is because Chat-GPT is a general-purpose model. However, the performance of some relatively small task-specific

models far exceeds ChatGPT, demonstrating the immense potential of transformers in mastering logical operation rules and the necessity of equipping the data-driven chatGPT with the logical rules for enhancing the performance on complex rule reasoning tasks such as math or coding.

Note that the EA of ChatGPT on Hard Ruletaker is slightly higher than on Ruletaker-3ext, this is because the labels in Hard Ruletaker are only True or False and we exclude data that ChatGPT considers Unknown (less than 10%). Though this may overestimate the performance, it does not affect our conclusion. Additionally, the EA of IBR on hard datasets is much higher than FaiRR and equal to Roberta. This is because IBR first predicts the final answer and then gives a reasoning process, and only the reasoning process is derived by stepwise backward inference.

5.2. Generalization to Higher Depths

In this section, we experiment with a setting where models are trained on reasoning depths less than or equal to 3 and tested on Ruletaker-D5 which contains problems that require reasoning up to depth 5. The reasoning depth are defined based on the minimal reasoning depth using the forward-chaining reasoning method (Tafjord et al., 2021). But we use resolution refutation which is different from forward-chaining in principle and thus leads to different minimal reasoning depth for the same instance. However, in a statistical sense, data with higher reasoning depth for forward-chaining is generally higher for resolution refutation. So it can also be a reference to compare the generalization ability of different methods using the depth defined by previous work.

From Table 2, we can find that the performance drop of GFaiR is smaller with the increasing reasoning depth. For example, considering the performance drops between d = 3 to d = 5, GFaiR has 1.6% drop in entailment accuracy. In contrast, FaiRR and NLProofs drop 14.4% and 24.5% in entailment accuracy, respectively. This indicates that our model's ability to generalize to higher reasoning

Model	5var	8var	10var	12var
T5	95.5	87.8	82.3	80.9
GFaiR	**95.5**	**91.3**	**90.1**	**89.4**

Table 4: Model performance on GRL dataset.

Model	16,21v	25,32v	35,48v	60,70v
T5	88.2	87.4	82.9	77.4
GFaiR	**93.6**	**92.4**	**91.7**	**91.3**

Table 5: Model performance on RCL dataset.

Model	Ruletaker-3ext		Hard Ruletaker*	
	EA	FA	EA	FA
FaiRR	99.0	98.4	41.1	39.8
FaiRR+	98.4	98.3	41.5	41.4
GFaiR-	97.5	97.2	72.4	68.6
GFaiR	98.1	98.0	73.9	71.7

Table 6: Results of ablation study.

depth is better.

5.3. In-domain Performance on Complex Reasoning Scenarios

To investigate the in-domain performance of different methods in complex reasoning scenarios, we evaluate different methods on Hard Ruletaker** dataset. Experimental results are shown in Table 3, from which we can find that compared to IBR, FaiRR, and NLProofs, GFaiR achieves better performance. Combining the experimental results in Table 1, we can conclude that GFaiR is more effective in handling complex logical reasoning scenarios by introducing resolution refutation.

5.4. Performance on Ruletaker-E

We also wish to see the performance on scenerios with implicitly expressed existence quantifiers. To do this, we evaluate different method's performances on the Ruletaker-E dataset. Experimental results are shown in Table 3, from which we can find that compared to FaiRR, IBR, and NLProofs, GFaiR achieves better performance, which indicates that it is also effective in handling implicitly expressed existence quantifiers by combining resolution refutation. Additionally, the difference between EA and FA also indicates that our model is faithful in scenarios with implicitly expressed existence quantifiers.

5.5. Performance on Natural Language Satisfiability Task

We further evaluate GFaiR on natural language satisfiability (NLSAT) task, whose aim is to determine whether there is a contradiction in the given NL Theory. In this task, we do not need the process of refutation, so the converter only needs to convert the NL Theory into natural language representations similar to the Skolem normal form, and then directly use our reasoning model to infer whether there is a contradiction in the given NL Theory.

Specially, there are two datasets available in this task, Grounded Rule Language (GRL) dataset and

Relative Clause Fragment (RCL) dataset. These datasets are more challenging compared to Hard RuleTaker (Richardson and Sabharwal, 2022). This is because these datasets demand the model to reason only based on rules and the number of reasoning steps required to solve the problem significantly exceeds that of Hard Ruletaker. So we use these datasets to further investigate the performance of our approach in more complex reasoning scenarios. Because there are no facts available on these datasets and models designed with forward or backward chaining rely on facts during inference. Therefore, we cannot apply these models to such tasks. Instead, we compare GFaiR with the T5-large two-stage fine-tuning method (Richardson and Sabharwal, 2022).

Experimental results are shown in Table 4 and 5, from which we can observe that GFaiR outperforms the baseline methods on both datasets. Consequently, GFaiR is capable of handling more complex reasoning scenarios by combining the step-wise inference method and resolution refutation.

5.6. Ablation Study

To respectively explore the effects of resolution refutation and validity contrastive loss-based verifier in our model, we consider the following ablations: 1) FaiRR+: add the validity contrastive loss-based verifier to the FaiRR model. So comparing FaiRR+ with GFaiR can show the impact of resolution refutation; 2) GFaiR-: replace the validity contrastive loss-based verifier with the verifier proposed by Yang et al. (2022) to check its impact.

Results on Ruletaker-3ext and Hard Ruletaker* datasets are given in table 6. From these results, we can know that even if adding a verifier to FaiRR, the performance on Hard Ruletaker* dataset is lower than GFaiR, which signifies the effectiveness of combining resolution refutation. Furthermore, we can know that GFaiR's performance is better than GFaiR-, which shows the effectiveness of the validity contrastive loss-based verifier.

6. Related Work

Natural Language Reasoning with First-Order Logic First-order logic has a wide range of cov-

erage. For example, it includes the majority of reasoning situations in commonsense reasoning (Davis, 2017). Additionally, it can represent most problems in mathematics and domains such as Euclidean geometry, making it widely used in automated theorem provers (Nawaz et al., 2019). As a result, FOL reasoning ability is a fundamental reasoning ability (Davis, 2017) widely used in existing reasoning benchmarks. For example, LogiQA (Liu et al., 2021) and ReClor (Yu et al., 2020) are two benchmarks widely used in logical reasoning. However, Tian et al. (2021) points out that FOL reasoning ability is not disentangled from other reasoning abilities such as commonsense reasoning in these benchmarks. So even if a model performs poorly on these datasets, it can't be concluded that the model lacks the reasoning ability. Starting from Clark et al. (2021), there are a series of novel benchmarks which measure logical reasoning independently of other forms of reasoning. We focus on these benchmarks to check our model's FOL reasoning ability. Since our method focuses on first-order logic reasoning based on natural language, it can easily be adapted to other forms of natural language-based reasoning problems.

Proof Generation One of our task's goals is to give a reasoning process, which is similar to the task of proof generation. Proof generation focuses on generating a reasoning chain from the given NL Theory to the conclusion, which aims at improving the model's interpretability (Rudin, 2019; Hase and Bansal, 2020). Recently, some works have been working on the problem of proof generation. Prover (Saha et al., 2020) trains a RoBERTa-based model that predicts nodes and edges of the proof graph. ProofWriter (Tafjord et al., 2021) is a T5-based model that iteratively generates one-hop conclusions and proofs from the NL Theory. FaiRR (Sanyal et al., 2022) further decomposes each reasoning step into selecting rules, selecting facts and reasoning based on selected rules and facts, which is similar to the reasoning process of forward reasoning. IBR (Qu et al., 2022) draws inspiration from backward reasoning and designs an iterative backward reasoning model. NU (Picco et al., 2021) also employs backward reasoning but it cannot generate a reasoning process. NLProofs (Yang et al., 2022) is also a stepwise reasoning method that using verifier-guided search. However, the validity contrastive loss-based verifier is more suitable for the reasoning scenerios of resolution. Another work MultiProver (Saha et al., 2021) aims at generating multiple proofs for a hypothesis.

7. Conclusion

In this paper, we propose GFaiR, a faithful and generalizable model capable of handling complex logical reasoning scenarios by introducing a validity contrastive loss-based verifier and resolution refutation. Experimental results also shows that GFaiR achieves better performance especially on Hard RuleTaker and Hard RuleTaker* datasets.

8. Acknowledgments

9. Bibliographical References

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

Samuel R Buss. 1998. An introduction to proof theory. *Handbook of proof theory*, 137:1–78.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2021. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3882–3890.

Antonia Creswell and Murray Shanahan. 2022. Faithful reasoning using large language models. *arXiv preprint arXiv:2208.14271*.

Ernest Davis. 2017. Logical formalizations of commonsense reasoning: A survey. *Journal of Artificial Intelligence Research*, 59:651–723.

Olga Golovneva, Moya Peng Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. 2023. ROSCOE: A suite of metrics for scoring step-by-step reasoning. In *The Eleventh International Conference on Learning Representations*.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Luke Benson, Lucy Sun, Ekaterina Zubova, Yujie Qiao, Matthew Burtell, et al. 2022. Folio: Natural language reasoning with first-order logic. *arXiv preprint arXiv:2209.00840*.

Peter Hase and Mohit Bansal. 2020. Evaluating explainable ai: Which algorithmic explanations help

users predict model behavior? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5540–5552.

Mehran Kazemi, Najoung Kim, Deepti Bhatia, Xin Xu, and Deepak Ramachandran. 2023. Lambada: Backward chaining for automated reasoning in natural language. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6547–6568.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2021. Logiqa: a challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3622–3628.

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

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. *arXiv preprint arXiv:2301.13379*.

Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi, and Yasuhiro Sogawa. 2023. Learning deductive reasoning from synthetic corpus based on formal logic. In *Proceedings of the 40th International Conference on Machine Learning,*, pages 25254–25274.

M Saqib Nawaz, Moin Malik, Yi Li, Meng Sun, and M Lali. 2019. A survey on theorem provers in formal methods. *arXiv preprint arXiv:1912.03028*.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. 2023. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint arXiv:2305.12295*.

Gabriele Picco, Thanh Lam Hoang, Marco Luca Sbodio, and Vanessa Lopez. 2021. Neural unification for logic reasoning over natural language. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3939–3950.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. 2022. Reasoning with language model prompting: A survey. *arXiv preprint arXiv:2212.09597*.

Hanhao Qu, Yu Cao, Jun Gao, Liang Ding, and Ruifeng Xu. 2022. Interpretable proof generation via iterative backward reasoning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2968–2981.

Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, Henghui Zhu, Rui Dong, Deguang Kong, Juliette Burger, Anjelica Ramos, zhiheng huang, William Yang Wang, George Karypis, Bing Xiang, and Dan Roth. 2023. STREET: A MULTI-TASK STRUCTURED REASONING AND EXPLANATION BENCHMARK. In *The Eleventh International Conference on Learning Representations*.

Kyle Richardson and Ashish Sabharwal. 2022. Pushing the limits of rule reasoning in transformers through natural language satisfiability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 11209–11219.

John Alan Robinson. 1965. A machine-oriented logic based on the resolution principle. *Journal of the ACM (JACM)*, 12(1):23–41.

Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215.

Stuart J Russell. 2010. *Artificial intelligence a modern approach*. Pearson Education, Inc.

Swarnadeep Saha, Sayan Ghosh, Shashank Srivastava, and Mohit Bansal. 2020. Prover: Proof generation for interpretable reasoning over rules. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 122–136.

Swarnadeep Saha, Prateek Yadav, and Mohit Bansal. 2021. multiprover: Generating multiple proofs for improved interpretability in rule reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3662–3677.

Soumya Sanyal, Harman Singh, and Xiang Ren. 2022. Fairr: Faithful and robust deductive reasoning over natural language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1075–1093.

Viktor Schlegel, Kamen Pavlov, and Ian Pratt-Hartmann. 2022. Can transformers reason in fragments of natural language? In *Proceedings of the 2022 Conference on Empirical Methods*

in Natural Language Processing, pages 11184–11199.

Changzhi Sun, Xinbo Zhang, Jiangjie Chen, Chun Gan, Yuanbin Wu, Jiaze Chen, Hao Zhou, and Lei Li. 2021. Probabilistic graph reasoning for natural proof generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3140–3151.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. Proofwriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. 2021. Diagnosing the first-order logical reasoning ability through logic-nli. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3738–3747.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, pages 24824–24837.

Kaiyu Yang, Jia Deng, and Danqi Chen. 2022. Generating natural language proofs with verifier-guided search. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 89–105.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, page 5754–5764.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. In *International Conference on Learning Representations*.

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang, and Guy Van den Broeck. 2023. On the paradox of learning to reason from data. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 3365–3373.

A. Dataset details

Due to the need for intermediate reasoning processes when training our model, we employed the FOL prover provided in the Stanford CS221 course page and Prover9 to automatically extract the reasoning process for these two types of tasks respectively. Below we describe our method to extract the reasoning process. Since the dataset is a synthetic dataset, regular expressions can be used to convert each data back to its corresponding FOL representation. Then we apply Prover9 or the FOL prover provided in the Stanford CS221 course page to each data and obtain its intermediate reasoning process of FOL representations. Finally, we transform the intermediate reasoning process from FOL representations into natural language representations by using natural language templates. Although this approach introduces some noise limited by the prover we used (redundant and excessively long reasoning steps), it does not hinder our model from achieving excellent generalization performance across all tasks.

Additionally, Richardson and Sabharwal (2022) illustrated that they found around 1% mismatched labels on the Ruletaker-3ext dataset. However, they only correct the train and dev set of the Ruletaker-3ext-sat dataset. As a result, we correct the test set of the Ruletaker-3ext dataset and the Ruletaker-D5 dataset for our experiments using the same method as Richardson and Sabharwal (2022).

B. Baselines details

B.1. ChatGPT Baseline

In order to automatically evaluate the accuracy of the reasoning process generated by ChatGPT, we need to know that the intermediate conclusions are derived from which theories are selected from the theory set. Therefore, we use the form of instruct+chain-of-thought prompt to strictly restrict its output form, specifically, we use 4-shot for Hard Ruletaker and 5-shot for Ruletaker-3ext-sat and Hard Ruletaker* because there is no label Unknown in Hard Ruletaker and we need one more example for the condition Unknown when testing on other datasets. However, there will still be a small portion of data (less than 10%) that we cannot parse the output of ChatGPT, so we exclude this portion of data. We only tested 3000 pieces of data in three datasets respectively using gpt-3.5-turbo due to cost reasons.

B.2. IBR Baseline

Since IBR targets the problems in the Close World Assumption, we made simple modifications to adapt to our target task. Specifically, the QA prediction module still first predicts the answer but we remove the strategy prediction module along with the strategy loss. This is because the search space of our target tasks is infinite so we can not

generate proof when the answer is Unknown and the strategy is always Proof. As a result, if the QA prediction module predicts Unknown, we stop and return the results. On the contrary, we will apply other modules in IBR to get the reasoning process. During training, all the modules of IBR are trained together with three types of losses: parent prediction loss, child prediction loss, and QA prediction loss (strategic prediction loss has been removed). However, when the gold answer is Unknown, there is no parent prediction loss and child prediction loss, which will introduce some noise. As a result, we implement the QA prediction module apart from two other modules.

C. implementation details

To reduce the search space and improve the inference efficiency of our model, we combine two complete inference strategies specifically designed for resolution refutation when experimenting with the natural language reasoning with first-order logic task, including set of support strategy and linear resolution strategy. However, when experimenting with the natural language satisfiability task, we can not combine these inference strategies because the task is different. In addition, we use beam search with beam size 5 for RuleTaker-E and 2 for other datasets.

Previous work (Buss, 1998) has shown that using linear resolution strategy and set of support strategy together will not affect the completeness of resolution refutation under first order logic. Set of support strategy requires that at least one of the two clauses involved in each resolution step is the negation of the inference target (hypothesis or the negation of the hypothesis) or a descendant of the negation of the inference target. Linear resolution strategy requires that one of the two clauses involved in each resolution step (except the first step) is the clause derived from the previous resolution step. Combining these two strategies, we can know that one of the two clauses involved in the first step is the negation of the inference target (from the set of support strategy), and one of the two clauses involved in the other steps is the clause derived from the previous resolution step (from the linear resolution strategy). From these we can know that one of the two clauses involved in each resolution step is determined. So we can remove the xlnet-based pre-selector while regarding that the pre-selector always choose the negation of the inference target in the first step, and choose the clause derived from the previous resolution step in the other resolution steps.