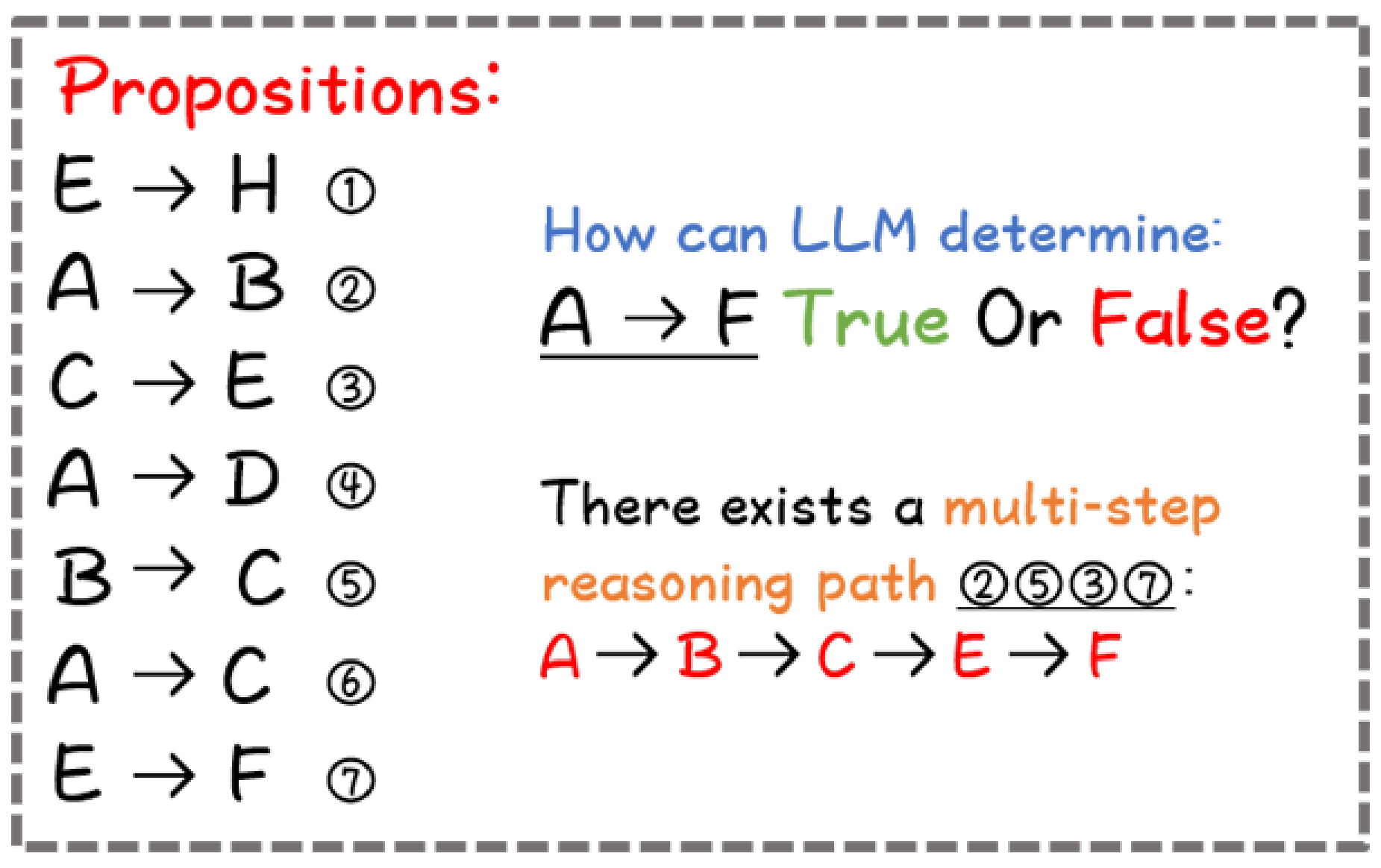
ChatLogic: Integrating Logic Programming with Large Language Models for Multi-Step Reasoning

Zhongsheng Wang, Jiamou Liu*, Qiming Bao, Hongfei Rong, Jingfeng Zhang
School of Computer Science, University of Auckland
zwan516@aucklanduni.ac.nz jiamou.liu@auckland.ac.nz

INTRODUCTION

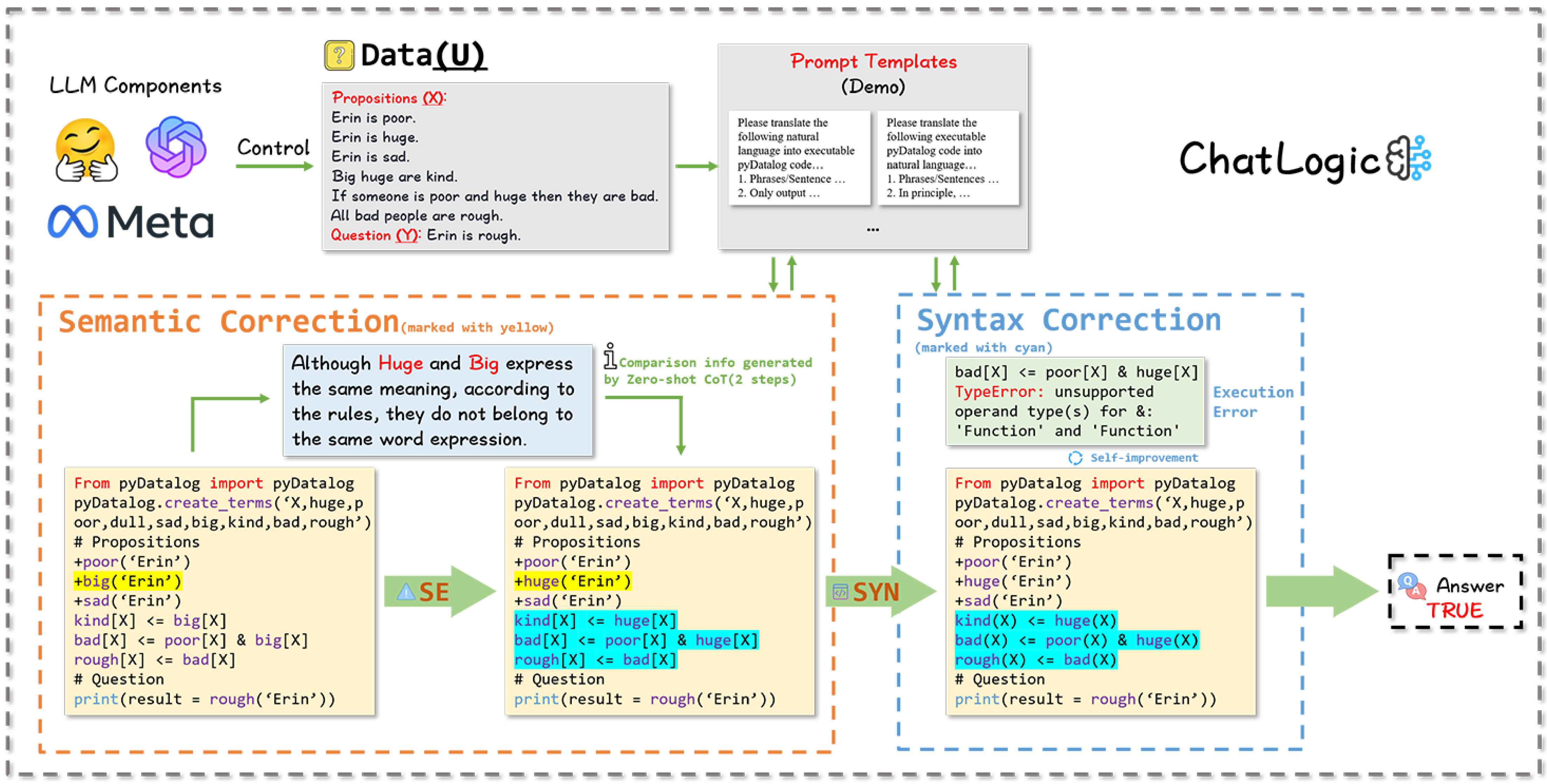
The performance of existing large language models (LLMs) for multi-step deductive reasoning is not ideal. Better performance often relies on a larger number of model parameters. Letting LLM capture key information from long and cluttered text is a key problem.



ChatLogic is an innovative framework that significantly improves the performance of LLMs like ChatGPT and GPT-4 in multi-step reasoning tasks by integrating logic programming. It leverages LLMs' situational understanding and imitation skills, and enhances multi-step deductive reasoning with symbolic memory. Our approach **transforms logic problems into symbolic integration** with an inference engine, leading to substantial improvements in LLMs' multi-step reasoning capabilities.

CHATLOGIC FRAMEWORK

ChatLogic enhances LLMs' reasoning through four stages: input processing, semantic and syntax correction, and local execution. It iteratively refines logic code, improving multi-step reasoning executability and accuracy. The process, highlighted by semantic and syntax self-corrections, boosts system performance, as depicted in the framework's diagram.



Propositions (X):
Erin is poor.
Erin is huge.
Erin is sad.
Big huge are kind.
If someone is poor and huge then they are bad.
All bad people are rough.
Question (Y): Erin is rough.

Please translate the following natural language into executable pyDatalog code…
1. Phrases/Sentence …
2. Only output …

Please translate the following executable pyDatalog code into natural language…
1. Phrases/Sentences …
2. In principle, …

Although Huge and Big express the same meaning, according to the rules, they do not belong to the same word expression.

```
From pyDatalog import pyDatalog
pyDatalog.create_terms('X,huge,poor,dull,sad,big,kind,bad,rough')
# Propositions
+poor('Erin')
+big('Erin')
+sad('Erin')
kind[X] <= big[X]
bad[X] <= poor[X] & big[X]
rough[X] <= bad[X]
# Question
print(result = rough('Erin'))
```

```
From pyDatalog import pyDatalog
pyDatalog.create_terms('X,huge,poor,dull,sad,big,kind,bad,rough')
# Propositions
+poor('Erin')
+huge('Erin')
+sad('Erin')
kind[X] <= huge[X]
bad[X] <= poor[X] & huge[X]
rough[X] <= bad[X]
# Question
print(result = rough('Erin'))
```

```
bad[X] <= poor[X] & huge[X]
TypeError: unsupported operand type(s) for &: 'Function' and 'Function'
```

```
From pyDatalog import pyDatalog
pyDatalog.create_terms('X,huge,poor,dull,sad,big,kind,bad,rough')
# Propositions
+poor('Erin')
+huge('Erin')
+sad('Erin')
kind(X) <= huge(X)
bad(X) <= poor(X) & huge(X)
rough(X) <= bad(X)
# Question
print(result = rough('Erin'))
```

ALGORITHM

Algorithm details ChatLogic's method for response querying, highlighting the LLM-driven sub-tasks, excluding local execution. It includes loops for semantic and syntax corrections, utilizing zero-shot CoT for semantic accuracy and introducing an upper limit on syntax corrections to prevent infinite loops. This balance enhances ChatLogic's robustness and effectiveness in multi-step reasoning tasks, despite slightly limiting inferential depth.

Algorithm 1 The Algorithm of ChatLogic

```
Input: U ← Rules supplemented based on the close-world assumption
X ← Proposition group (contains facts and rules)
Y ← Question
Output: TRUE/FALSE (Answer to Y given U, X)
DifferentFlag = TRUE
                                                         ▷ Semantic Correction
while DifferentFlag do
    Code ← PropositionTransformation($X, Y, U$)    ▷ Generate logic program based on close-world assumption
    RevProposition ← ReverseTransformation(Code, U)    ▷ Convert code back to natural language
    DifferentInfo ← TextComparison(($X, Y$), RevProposition)
    DifferentFlag ← JudgeInfo(DifferentInfo)    ▷ Determining semantic similarity status with zero-shot CoT in 2 steps
end while
                                                         ▷ Syntax Correction
ExecutionError = NULL                                    ▷ Execution result record
while Code cannot be executed do
    Code ← CodeImprovement(Code, ExecutionError)         ▷ Improve code based on error info
    if Running Time Overflow then
        Terminate WHILE Loop
    end if
end while
return CodeExecution(Code)                 ▷ Get results by executing pyDatalog code locally
```

CHATGPT VS. CHATLOGIC(CHATGPT)

Propositions:
Facts:
Anne is huge. Anne is strong. Fiona is thin.
Fiona is little. Alan is clever. Dave is rough. Dave is bad.
Rules:
If someone is not big then they are rough.
(Continue on the right)

If someone is not sad then they are kind.
If someone is clever then they are quiet.
If someone is quiet and not poor then they are smart.
If someone is rough and not big then they are dull.
If someone is dull then they are small.
Question:
Anne is tiny?

ChatGPT:
To determine whether "Anne is tiny" is correct, let's examine the given propositions step-by-step and follow the chain of reasoning.
From: Anne is huge.
Based on Proposition 8: "If someone is not big then they are rough. **"Since Anne is huge, she is big."** Hence, she is not rough.
We need to determine if Anne can be categorized as small and tiny.
(WRONG! " Huge " is not the same as "big ". We should not infer downward based on " Anne is big ", but based on the closed-world assumption: The proposition: "Anne is big" is not clearly defined, we should think that Anne is not big.)
Therefore, we don't have enough evidence to say that "Anne is tiny" based on the given propositions.
Answer: 0 (The statement "Anne is tiny" is wrong based on the provided propositions.)

ChatLogic:

```
# Declare the pyDatalog variables
pyDatalog.create_terms('X,huge,strong,little,rough,clever,sad,kind,quiet,tiny,big […]')
# Define the facts
+huge('Anne')
+strong('Anne')
+little('Fiona') […]
# Define the rules
rough(X) <= ~big(X)
kind(X) <= ~sad(X)
quiet(X) <= clever(X) […]
# Query the knowledge base
result = tiny('Anne')
if result:
    print(1)
else:
    print(0)
```

```
result: 1
Anne is tiny.
```

EXPERIMENTS

Experiment 1: Performance Evaluation.

We compare three synthetic multi-step reasoning QA datasets: PARARULE-Plus, CONCEPTRULES V1 & V2. Examine the baseline capabilities of LLMs, the performance improvement after applying specific prompt items for zero-shot CoT, and the effect of embedding

Model	Method	Depth=2	Depth=3	Depth=4	Depth=5	Total
GPT-3.5	Base	0.4	0.34	0.32	0.3	0.344
	Zero-shot CoT	0.42	0.42	0.41	0.3	0.3875
	ChatLogic	**0.49**	**0.56**	**0.65**	**0.41**	**0.5275**
GPT-4	Base	0.65	0.75	0.42	0.4	0.555
	Zero-shot CoT	**0.72**	0.72	0.62	**0.7**	0.69
	ChatLogic	**0.72**	**0.8**	**0.7**	**0.7**	**0.73**
Llama 2-7B	Base	0.11	0.06	0.01	0.01	0.0475
	Zero-shot CoT	0.15	**0.13**	0.08	0.06	0.105
	ChatLogic	**0.2**	**0.13**	**0.22**	**0.18**	**0.1825**

Results on PARARULE-Plus dataset

Model	Method	CONCEPTRULES V1		CONCEPTRULES V2	
		simplified	full	simplified	full
GPT-3.5	Base	0.57	0.55	0.5	0.51
	Zero-shot CoT	0.63	0.51	0.7	0.67
	ChatLogic	**0.69**	**0.67**	**0.79**	**0.74**
GPT-4	Base	0.95	0.94	0.89	0.86
	Zero-shot CoT	**0.96**	**0.97**	**0.95**	**0.94**
	ChatLogic	**0.96**	0.96	0.94	**0.94**
Llama 2-7B	Base	0.32	0.29	0.31	0.24
	Zero-shot CoT	0.42	0.41	0.33	0.3
	ChatLogic	**0.48**	**0.49**	**0.37**	**0.36**

Results on CONCEPTRULES V1 & V2 dataset

Result: The ChatLogic framework integrating LLMs has greatly improved the inference accuracy, most of which exceeds the baseline and zero-shot CoT methods.

Experiment 2: Ablation Study.

In order to separately test the capabilities of the two correction modules in the framework, we split them and verified their performance effects respectively. As the modules continue to be added, we expect a result in which the accuracy will gradually improve.

Dataset	Model	Base	SE	SE+SYN
CONCEPTRULES V1	GPT-3.5	0.63	0.68	0.7
	GPT-4	0.92	0.96	0.96
	Llama 2-7B	0.31	0.60	0.62
CONCEPTRULES V2	GPT-3.5	0.6	0.73	0.8
	GPT-4	0.92	0.93	0.95
	Llama 2-7B	0.33	0.52	0.53
PARARULE-Plus	GPT-3.5	0.26	0.5	0.62
	GPT-4	0.54	0.64	0.7
	Llama 2-7B	0.1	0.16	0.16

Results for ablation study on 3 datasets

Result: Both modules, Semantic Correction (SE) and Syntax Correction (SYN), contribute to the accuracy improvement of multi-step reasoning. However, for models with relatively weak performance, the performance of the grammar correction link may not be ideal.