ABSTRACT

Large language models (LLMs) have revolutionized many areas (e.g., natural language processing, software engineering, etc.) by achieving state-of-the-art performance on extensive downstream tasks. Aiming to achieve robust and general artificial intelligence, there has been a surge of interest in investigating the reasoning ability of the LLMs. Whereas the textual and numerical reasoning benchmarks adopted by previous works are rather shallow and simple, it is hard to conclude that the LLMs possess strong reasoning ability by merely achieving positive results on these benchmarks. Recent efforts have demonstrated that the LLMs are poor at solving sequential decision-making problems that require common-sense planning by evaluating their performance on the reinforcement learning benchmarks. In this work, we conduct an in-depth assessment of several state-of-the-art LLMs’ reasoning ability based on the inductive logic programming (ILP) benchmark, which is broadly recognized as a representative and challenging measurement for evaluating logic program induction/synthesis systems as it requires inducing strict cause-effect logic to achieve robust deduction on independent and identically distributed (IID) and out-of-distribution (OOD) test samples. Our evaluations illustrate that compared with the neural program induction systems which are much smaller in model size, the state-of-the-art LLMs are much poorer in terms of reasoning ability by achieving much lower performance and generalization using either natural language prompting or truth-value matrix prompting1.

KEYWORDS
Large Language Models, Relational Reasoning, Program Induction

1 INTRODUCTION

Large language models (LLMs) have achieved great breakthroughs in various domains such as natural language processing [2, 28], software engineering [7, 16], finance [32, 33], etc. There has been a recent increase in interest in exploring the reasoning ability of LLMs, which is regarded as a crucial ability of Artificial General Intelligence (AGI) [17, 20, 31]. Prevalent reasoning ability evaluation benchmarks adopted by previous literature include arithmetic [5, 23], symbolic reasoning [25, 31], commonsense [24, 26] etc. And with the recent advancement of the in-context few-shot (zero-shot) prompting techniques [2, 19, 31], LLMs manage to achieve state-of-the-art few-shot (zero-shot) learning performance on these benchmarks without training. Concisely, in-context few-shot (zero-shot) prompting refers to the techniques that provide input to the
2 PRELIMINARIES

2.1 Inductive Logic Programming

Program induction & program synthesis [1, 8, 9] are tasks that aim to induce a program that satisfies a given specification. Program induction requires the model to induce the program implicitly and conduct inference directly by the model itself without executing an explicit program, while program synthesis requires the model to generate an explicit program and execute it for inference. Inductive logic programming (ILP) [11, 18, 22] is a sub-field in which specification comes in the form of input-output examples. It requires the model to express the program with first-order logic rules. ILP has long been considered a task that is demanding in logical (relational) reasoning ability to accomplish. Specifically, given a set of background predicates \( B \) (also called premise), a set of positive examples \( \rho \), and a set of negative samples \( \eta \), the goal of a program induction/synthesis model is to derive a logic program \( C \) that satisfies the following two conditions: (1) based on the premise \( B \), \( C \) entails all the positive examples, denoted by \( \forall \rho : B, C \models \rho \), and (2) based on the premises, \( C \) does not entail any of the negative examples, denoted by \( \forall \eta : B, C \not\models \eta \).

Differentiable logic machines (DLM) [34] model is a state-of-the-art neural program induction model for the ILP tasks that realizes first-order logic (FOL) rules in a neural manner. Specifically, the DLM model uses neural networks as computation units that implement "soft" logic operators. Then based on the computation units, the DLM model can approximate the forward-chaining mechanism of FOL by stacking multiple layers of computation units. It takes the truth value matrices of the background predicates as input and outputs the truth value matrix of the target predicate. With the strong inductive bias of the model design, DLM has been proven to achieve superior performance and generalization on benchmarks that demand strong reasoning ability such as relational reasoning and decision-making tasks.

2.2 Large Language Models

Large language models (LLMs) denote the pre-trained Transformer architecture-based AI models. By leveraging large amounts of multimodal data and the pre-training & fine-tuning learning techniques, the LLMs are reported to achieve state-of-the-art performance on many downstream tasks (e.g. machine translation, numerical reasoning, code clone detection, etc.). Specifically, the GPT model family (e.g. GPT-3.5, GPT-4, etc.) are pre-trained with an autoregressive language modeling objective. The formal definition of the autoregressive loss \( L_{\text{aut}} \) is as follows:

\[
L_{\text{aut}} = -\sum_{t=2}^{n} \log p(y_t|y_{t-1},...,y_1)
\]  

Given a sentence of \( n \) tokens, the model is trained to maximize the likelihood of the ground-truth token \( y_t \) of the current time step \( t \) based on its previous sequence \( y_{t-1},...,y_1 \). In-context few-shot (zero-shot) prompting is a recently proposed technique that conditions the LLM with some initial input to improve performance. The input can be a few examples (few-shot) [3, 31] or instructions regarding the task (zero-shot) [30].
generates random data points represented in the form of truth-value matrix prompting (Section 3.2). Then the details of the (tv, nl)-modality compiler are introduced. It transforms truth value prompting into corresponding natural language prompting to evaluate the models’ relational reasoning ability using different data modalities (Section 3.3). Finally, we illustrate the evaluation module which allows measuring models’ results from multiple aspects (Section 3.4).

3.1 Relational Reasoning Benchmark
There are many available relational reasoning benchmarks adopted by previous literature, e.g., family tree reasoning [9, 10], general graph reasoning [9, 13, 34], relational queries [27], etc. In this work, we conduct experiments on two broadly used relational reasoning benchmarks that are adopted by previous state-of-the-art neural program induction models [9, 10, 21, 34]: family tree reasoning and general graph reasoning.

Family tree reasoning. The family tree reasoning benchmark consists of tasks that require the model to induce programs that deduce more complex relations based on some basic properties of family members or relations between them. Specifically, a family tree is represented with four basic predicates: IsMother(x, y), IsSon(x, y), IsDaughter(x, y). E.g. IsMother(x, y) is True if y is x’s mother, the semantics of the other basic predicates are similar. This benchmark contains 5 target predicates to induce. The following shows the details of the HasSister and IsGrandparent tasks:

- **HasSister(x):** the semantics of this predicate is to determine whether x has a sister. The ground-truth program to induce is:
  \[
  \text{HasSister}(x) \iff \exists y, z, \text{IsDaughter}(y, z) \land \text{IsMother}(x, z)
  \]

- **IsGrandparent(x, y):** the semantics of this predicate is to determine whether y is the grandparent of x. The ground-truth program to induce is:
  \[
  \text{IsGrandparent}(x, y) \iff \exists z, ((\text{IsSon}(y, z) \land \text{IsFather}(x, z)) \\
  \lor (\text{IsDaughter}(y, z) \land \text{IsMother}(x, z)))
  \]

General graph reasoning. The general graph reasoning benchmark consists of tasks that require the models to infer the logic of high-level target predicates that describe properties/relations of a graph based on a basic predicate: HasEdge(x, y) (i.e., whether there is an undirected edge between node x and y in the graph). This benchmark contains 4 target predicates to infer. The following shows the details of the 4-Connectivity and 1-Outdegree tasks:

- **4-Connectivity(x, y):** the semantics of 4-Connectivity(x, y) is to determine whether there exists a path between node x and node y within 4 edges. The ground-truth program to induce is:
  \[
  \text{4-Connectivity}(x, y) \iff \exists z, (\text{HasEdge}(x, y) \lor \text{Invented}(x, y) \lor (\text{Invented}(x, z) \land \text{HasEdge}(z, y)) \lor (\text{Invented}(x, z) \land \text{Invented}(z, y)))
  \]

- **1-Outdegree(x):** the semantics of this predicate is to determine whether the outdegree of node x in a graph is exactly 1. The ground-truth program to induce is:
  \[
  \text{1-Outdegree}(x) \iff \exists y, \forall z, (\text{HasEdge}(x, y) \land \neg \text{HasEdge}(x, z))
  \]

3.2 Sample Generator
Given the specifications of each relational reasoning benchmark, the sample generator implements a truth value generator tool. Specifically, when it is called, the truth value generator tool generates the truth value matrices that represent the input & output relations of the desired program to be induced. Figure 2 shows a concrete truth value table representation of a HasSister task sample. It represents a family with four members \{P0, P1, P2, P3\}. P0 is the mother, P1 is the father, P2 and P3 are the two daughters of them. The input prompting therefore contains the truth value matrices of the input basic predicate IsMother(x, y), IsDaughter(x, y). E.g. for the IsMother(x, y), since P1 is the mother of both P2 and P3, the items within the matrix that represent IsMother(P2, P1) and IsMother(P3, P1) equal to 1. Then, given the input representation, the models are required to deduce the truth value matrix of the desired target predicate HasSister.

3.3 (tv, nl)-Modality Compiler
As the LLMs are natural language models whose primary training data source is natural language, we aim to evaluate their performance when the samples are represented in the form of natural language. Therefore, we implement a (tv, nl)-modality compiler to convert the sample from truth value prompting to natural language.
You are an agent who determines the relations in a family. I will first show you some examples, and then I will give you a new question for you to answer. The examples are shown as follows.

**Question:** For a family tree containing 6 family members, which is depicted with 4 kinds of relations: father, mother, son, daughter. The relations are: P0 is P2’s father; P0 is P3’s father; P1 is P2’s mother; P1 is P3’s mother; P2 is P0’s daughter; P2 is P1’s daughter; P3 is P0’s daughter; P3 is P1’s daughter. For a new relation sister, given the above-given facts, who has the relation sister? You must give the final answer in this format: Who has sister.

**Answer:** P2 has sister; P3 has sister.

Now this is the question for you to answer.

**Question:** For a family tree containing 5 family members, which is depicted with 4 kinds of relations: father, mother, son, daughter. The relations are: P1 is P0’s father; P2 is P4’s father; P4 is P0’s mother; P3 is P4’s mother; P0 is P1’s son; P0 is P4’s son; P4 is P2’s daughter; P4 is P3’s daughter. For a new relation sister, from the above-given facts, who has relation sister? You must give the final answer in this format: Who has sister.

**Answer:** P2 has sister; P3 has sister.

Figure 3: Natural language prompting of a HasSister task.

3.4 Evaluation Module

We implement the evaluation module for evaluations of results from both the truth value source and natural language source. This module allows evaluating test performance under the independent and identically distribution (IID) setting and generalization under the out-of-distribution (OOD) setting. Specifically, for the family tree benchmark, IID is defined as the test sample whose number of family members is the same as that of the training sample. OOD is defined as the test sample with a larger family size than that of the training sample. Similarly, for the general graph reasoning benchmark, IID refers to the test sample with the same number of graph nodes as the training data while OOD refers to those with a larger number of graph nodes.

4 EXPERIMENTAL SETTINGS

4.1 Data setup.

We illustrate the details of the data setup of the two benchmarks used in our study. For all the tasks of the family tree reasoning benchmark, the number of family members for the few-shot learning (training) data is set to 10. We use data samples with the same number of family members (n = 10) for IID performance evaluation; and samples with a larger number of family members (n = 20) for OOD generalization evaluation. Similarly, for the general graph reasoning benchmarks, the number of nodes of each graph for the few-shot learning is set to 10. The number of nodes of each IID test sample is the same as the training samples (n = 10), and each OOD test graph contains 20 (n = 20) nodes. For standard prompting, the number of training samples is set to 10, i.e., 10 families/graphs. The numbers of IID and OOD test samples are set to 10. For chain-of-thought prompting, due to the limitation of tokens, the number of training samples is set to one. The numbers of IID and OOD test samples are set to 10. As an example, for the IID test, 10 queries are conducted, each containing all the training samples and one unique testing sample. For the truth value prompting, we use Python list to represent the matrices. We conduct all experiments on a server with 48 cores of Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz, 4 Quadro RTX 8000 GPUs, and 256G RAM.
Table 1: Performance and generalization results of the LLMs using standard natural language prompting and DLM using few-shot training. Red number denotes the best performance/generalization result among all the LLMs for a task, number in the grey box denotes the best result among all the evaluated baselines.

<table>
<thead>
<tr>
<th>Task</th>
<th>GPT-4 Turbo</th>
<th>GPT-4</th>
<th>GPT-3 Turbo</th>
<th>llama-7b</th>
<th>llama-13b</th>
<th>DLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>n=10</td>
<td>n=20</td>
<td>n=10</td>
<td>n=20</td>
<td>n=10</td>
<td>n=20</td>
<td>n=10</td>
</tr>
<tr>
<td>HasFather</td>
<td>100.00%</td>
<td>99.17%</td>
<td>100.00%</td>
<td>99.60%</td>
<td>61.34%</td>
<td>46.87%</td>
</tr>
<tr>
<td>HasSister</td>
<td>74.17%</td>
<td>81.41%</td>
<td>70.61%</td>
<td>81.18%</td>
<td>46.89%</td>
<td>54.96%</td>
</tr>
<tr>
<td>IsGrandparent</td>
<td>57.00%</td>
<td>35.42%</td>
<td>45.83%</td>
<td>13.98%</td>
<td>32.37%</td>
<td>9.54%</td>
</tr>
<tr>
<td>IsUncle</td>
<td>26.67%</td>
<td>17.08%</td>
<td>49.05%</td>
<td>10.15%</td>
<td>2.86%</td>
<td>2.15%</td>
</tr>
<tr>
<td>IsMGUncle</td>
<td>10.00%</td>
<td>10.00%</td>
<td>48.33%</td>
<td>16.67%</td>
<td>0.00%</td>
<td>3.33%</td>
</tr>
<tr>
<td>4-Connectivity</td>
<td>62.43%</td>
<td>11.77%</td>
<td>73.28%</td>
<td>12.08%</td>
<td>44.25%</td>
<td>6.29%</td>
</tr>
<tr>
<td>6-Connectivity</td>
<td>63.01%</td>
<td>1.43%</td>
<td>73.98%</td>
<td>37.85%</td>
<td>25.67%</td>
<td>6.08%</td>
</tr>
<tr>
<td>1-Outdegree</td>
<td>100.00%</td>
<td>100.00%</td>
<td>57.77%</td>
<td>64.47%</td>
<td>3.33%</td>
<td>4.00%</td>
</tr>
<tr>
<td>2-Outdegree</td>
<td>88.57%</td>
<td>86.59%</td>
<td>86.67%</td>
<td>75.74%</td>
<td>6.67%</td>
<td>0.00%</td>
</tr>
</tbody>
</table>

**Figure 4:** An input-conflicting reasoning case of GPT-4.

**Figure 5:** A common sense-conflicting reasoning case of the GPT-4 Turbo model.

5 EVALUATION RESULTS

In this section, we study the following research questions:

- **RQ1:** How good is LLMs’ relational reasoning ability with standard natural language prompting?
- **RQ2:** How good is LLMs’ relational reasoning ability with truth value prompting?
- **RQ3:** Is the state-of-the-art prompting technique effective in boosting LLMs’ relational reasoning ability?

### 5.1 RQ1: How good is LLMs’ relational reasoning ability with standard natural language prompting?

Table 1 shows the in-context few-shot prompting results of the LLMs using standard natural language prompting and the few-shot training results of the DLM. Overall, we have the following findings. Among all the evaluated LLM models, GPT-4 Turbo and GPT-4 perform the best in terms of both test performance and OOD
generalization. Their superiority can be attributed to the fact that the two models are much larger in size and have access to a much larger amount of pretraining data than other LLMs baselines. Furthermore, we observe that although the LLMs can achieve great performance/generalization on relatively easier tasks, their results compromise significantly on tasks that require more complex task-solving logic (i.e. tasks that contain more predicates in the ground-truth program). For example, while the GPT-4 model achieves 100% F1-score in terms of performance on the HasFather task, it only achieves 49.05% on the IsUncle task. To better understand why the LLMs perform poorly on harder tasks, we conduct two case studies of the GPT-4 and GPT-4 Turbo models on the IsUncle test sample, shown in Figure 4 and Figure 5. For Figure 4, though it is stated in the input prompt that "P8 is P3’s mother; P3 is P8’s daughter" (bold blue text), GPT-4 generates an input-conflicting reasoning process stating that "P8 is the father of P3 and also the mother of P3" (bold red text). For Figure 5, as shown in GPT-4 Turbo’s answer, the model generates a common sense-conflicting reasoning process that claims that P0 and P5 are each other’s uncle (bold red text). These hallucination phenomena would therefore undermine the model’s relational reasoning ability, leading to erroneous conclusions.

Finally, though the DLM model is much smaller in size compared to the LLMs (DLM has only 60k parameters, while the smallest evaluated LLM has 7 billion parameters (llama-7B)), it manages to achieve the best results with only few-shot training. Note that the hardware requirement required for training the DLM model is much lower than even loading the LLMs (e.g. it only requires 429 MiB of GPU memory for training DLM while 36,957 MiB is required for loading llama-7B). The reason is that the logical reasoning knowledge injected in the DLM’s model design allows it to conduct robust program induction which makes it suitable for the relational reasoning tasks.

Answer to RQ1: GPT-4 and GPT-4 Turbo models present the strongest relational reasoning ability among all the evaluated LLMs. However, the LLMs’ performance/generalization drops significantly on tasks that require complex task-solving logic. Besides, the relational reasoning ability of the LLMs’ in-context few-shot prompting is generally worse than the few-shot trained DLM model which is much less hardware-demanding.

5.2 RQ2: How good is LLMs’ relational reasoning ability with truth value prompting?

We further assess the relational reasoning ability of the LLMs when using the same data modality as the DLM model (i.e. representing input/output predicate relations with truth value matrices). Table 2 shows the in-context few-shot learning results of the LLMs using truth value prompting. The result is marked as N/A if the model cannot generate the complete output array. We omit the results of the GPT-4, llama-7b, and llama-13b models as all their results are N/A. The reason for the complete N/A results for these three models is because they are relatively small in terms of context window whereas the GPT-4 Turbo and GPT-3.5 Turbo are capable of processing long context window and generating long output sequence.

Interestingly, we observe that the results of GPT-4 Turbo and GPT-3.5 Turbo on the family tree reasoning tasks decrease as compared to the corresponding standard natural language prompting results. This is because the natural language pretraining data regarding family relations is common and sufficient which allows the LLMs to perform well with natural language prompting on the family tree reasoning tasks. While the truth value pretraining data is much fewer comparatively which makes it challenging for the LLMs to reason well with truth value prompting. Whereas interestingly, the IID performance is consistent or even improves on the general graph reasoning tasks as compared to the standard natural language prompting results, the GPT-4 Turbo even manages to achieve the best performance and generalization on the 2-outdegree task. We believe that LLMs have the ability to perform graph reasoning with truth value matrices, which makes them promising for truth value-related tasks (e.g. logic synthesis [15]). We leave the further investigation to future work.

Answer to RQ2: LLMs with small context window are incapable of conducting relational reasoning with truth value prompting, while the LLMs with large context window present consistent or even improved IID performance on the general graph reasoning tasks compared to the standard natural language prompting results.

5.3 RQ3: Is the state-of-the-art prompting technique effective in boosting LLMs’ relational reasoning ability?

We further evaluate whether the recently proposed chain-of-thought (CoT) [31] prompting can improve the LLMs’ relational reasoning ability. Concretely, each exemplar in few-shot prompting is augmented with the direct cause-effect logic that represents the deduction process of the answer. Figure 6 shows an example CoT prompting of a HasSister sample, the bold text represents the example

Table 2: Performance and generalization results of DLM and LLMs using truth table prompting.

<table>
<thead>
<tr>
<th>Task</th>
<th>GPT-4 Turbo</th>
<th>GPT-3.5 Turbo</th>
<th>DLM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n=10</td>
<td>n=20</td>
<td>n=10</td>
</tr>
<tr>
<td>HasFather</td>
<td>79.17%</td>
<td>76.99%</td>
<td>47.57%</td>
</tr>
<tr>
<td>HasSister</td>
<td>47.06%</td>
<td>50.49%</td>
<td>38.41%</td>
</tr>
<tr>
<td>IsGrandparent</td>
<td>95.92%</td>
<td>N/A</td>
<td>6.62%</td>
</tr>
<tr>
<td>IsUncle</td>
<td>6.67%</td>
<td>N/A</td>
<td>0.00%</td>
</tr>
<tr>
<td>IsMutualUncle</td>
<td>0.00%</td>
<td>N/A</td>
<td>0.00%</td>
</tr>
<tr>
<td>4-Connectivity</td>
<td>60.76%</td>
<td>N/A</td>
<td>53.35%</td>
</tr>
<tr>
<td>6-Connectivity</td>
<td>65.77%</td>
<td>N/A</td>
<td>51.67%</td>
</tr>
<tr>
<td>1-Outdegree</td>
<td>98.90%</td>
<td>99.19%</td>
<td>50.05%</td>
</tr>
<tr>
<td>2-Outdegree</td>
<td>100.00%</td>
<td>95.31%</td>
<td>42.65%</td>
</tr>
</tbody>
</table>

Note that for all the evaluated tasks, the ground-truth output truth value matrices required to generate never exceed the maximum output token length of any LLMs.
You are an agent who determines the relations in a family. I will first show you some examples, and then I will give you a new question for you to answer. The examples are shown as follows. Now this is the question for you to answer.

**Question:** For a family tree containing 6 family members, which is depicted with 4 kinds of relations: father, mother, son, daughter. The relations are: P3 is P0’s father; P3 is P2’s father; P5 is P0’s mother; P5 is P2’s mother; P0 is P3’s daughter; P2 is P5’s daughter; P2 is P5’s daughter. For a new relation sister, from the above-given facts, who has relation sister? You must give the reasoning process and you must give the final answer in this format: Who has sister.

**Answer:** For P0: Since P5 is P0’s mother and P2 is P5’s daughter, P0 has 1 sister. Therefore, we can derive that P0 has sister. For P2: Since P5 is P2’s mother and P0 is P5’s daughter, P2 has 1 sister. Therefore, we can derive that P2 has sister.

Now this is the question for you to answer.

**Question:** For a family tree containing 6 family members, which is depicted with 4 kinds of relations: father, mother, son, daughter. The relations are: P0 is P1’s father; P5 is P1’s mother; P1 is P0’s daughter; P1 is P5’s daughter. For a new relation sister, from the above-given facts, who has relation sister? You must give the reasoning process and you must give the final answer in this format: Who has sister.

**Answer:** For P0: Since P5 is P0’s mother and P2 is P5’s daughter, P0 has 1 sister. Therefore, we can derive that P0 has sister. For P2: Since P5 is P2’s mother and P0 is P5’s daughter, P2 has 1 sister. Therefore, we can derive that P2 has sister.

Table 3: Performance and generalization results of DLM and LLMs using chain-of-thought prompting.

<table>
<thead>
<tr>
<th>Task</th>
<th>GPT-4-Turbo n=10</th>
<th>GPT-4 n=10</th>
<th>GPT-3.5 Turbo n=10</th>
<th>llama-7b n=10</th>
<th>llama-13b n=10</th>
<th>DLM n=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasFather</td>
<td>100.00%</td>
<td>100.00%</td>
<td>76.32%</td>
<td>59.49%</td>
<td>86.98%</td>
<td>100.00%</td>
</tr>
<tr>
<td>HasSister</td>
<td>86.39%</td>
<td>76.26%</td>
<td>79.88%</td>
<td>57.94%</td>
<td>52.91%</td>
<td>100.00%</td>
</tr>
<tr>
<td>IsGrandparent</td>
<td>63.57%</td>
<td>38.38%</td>
<td>70.32%</td>
<td>2.99%</td>
<td>7.08%</td>
<td>100.00%</td>
</tr>
<tr>
<td>IsUncle</td>
<td>44.44%</td>
<td>23.84%</td>
<td>96.67%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>85.00%</td>
</tr>
<tr>
<td>IsMother</td>
<td>10.00%</td>
<td>10.00%</td>
<td>40.00%</td>
<td>2.00%</td>
<td>0.00%</td>
<td>55.24%</td>
</tr>
<tr>
<td>4-Connectivity</td>
<td>26.00%</td>
<td>3.44%</td>
<td>79.78%</td>
<td>40.82%</td>
<td>39.33%</td>
<td>80.82%</td>
</tr>
<tr>
<td>6-Connectivity</td>
<td>54.82%</td>
<td>3.50%</td>
<td>86.31%</td>
<td>50.88%</td>
<td>35.55%</td>
<td>83.26%</td>
</tr>
<tr>
<td>1-Outdegree</td>
<td>86.39%</td>
<td>83.18%</td>
<td>38.19%</td>
<td>24.81%</td>
<td>14.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>2-Outdegree</td>
<td>100.00%</td>
<td>92.24%</td>
<td>76.67%</td>
<td>28.71%</td>
<td>20.67%</td>
<td>90.00%</td>
</tr>
</tbody>
</table>

Figure 6: Chain-of-thought prompting of a HasSister sample.

CoT prompting can even significantly decrease the LLMs’ performance/generalization for some tasks.

## 6 RELATED WORKS

### 6.1 LLMs for logical reasoning

With the rapid advancement of LLMs, there has been a recent surge in research that leverages LLMs to solve logical reasoning tasks, such as common-sense reasoning [24, 26], arithmetic reasoning [5, 23], symbolic reasoning [25], etc. It is observed in previous works that the standard prompting (only questions are given) is not sufficient as the model performs poorly [31]. Aiming to improve the LLMs’ performance, Brown et al. [3] propose prompting the model with a few input–output exemplars demonstrating the task, which is proven to be effective for a range of simple question-answering tasks. Wei et al. [31] propose providing the model with the concrete chain-of-thought prompting (i.e. I/O examples with the corresponding explicit reasoning steps) to the LLMs. It is demonstrated that the chain-of-thought prompting technique improves performance on arithmetic reasoning, commonsense reasoning, and symbolic reasoning tasks.

### 6.2 Neural Program Induction & Synthesis

Leveraging neural networks for program induction and program synthesis has become increasingly popular. Recent work has demonstrated the effectiveness of neural network-based program induction and synthesis methods in relational reasoning tasks with better noise tolerance and less manual effort [8, 10, 12]. Dong et al. [9] propose Neural Logic Machines (NLMs) which approximate logic predicates and logic rules with neural modules and achieves state-of-the-art performance on the program induction tasks. To improve the interpretability of the NLM model, Zimmer et al. [34] propose using soft logic operators as the computation unit which allows the model to generate interpretable logic programs. Cao et al. [4] propose a sketch-based hierarchical program synthesis framework for complex sequential decision-making problems based on dILP, which achieves high performance and generalizability.
7 CONCLUSION
In this work, we propose a general and extensible relational reasoning ability assessment pipeline for the LLMs and the neural program induction models. Based on the pipeline, we conduct the first comprehensive evaluations of the LLMs’ relational reasoning ability. Our results show that when using standard natural language prompting, the relational reasoning ability of LLMs’ in-context few-shot prompting is generally far from satisfaction compared with the program induction model which is much smaller in size. While the LLMs with large context window present consistent or even improved IID performance on the general graph reasoning tasks. And we further show that the current state-of-the-art prompting technique is not generally effective for improving LLMs’ relational reasoning ability.

8 ACKNOWLEDGEMENT
This research is supported by the National Research Foundation, Singapore, and the Cybersecurity Agency of Singapore. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and Cyber Security Agency of Singapore.

REFERENCES