AAAI-24 Symposium Spring Symposium on
User-Aligned Assessment of Adaptive AI Systems

Can LLMs translate SATisfactorily?

Assessing LLMs in Generating and Interpreting Formal Specifications

Rushang Karia, Daksh Dobhal, Daniel Bramblett, Pulkit Verma, Siddharth Srivastava

Presenter

AAIR
Autonomous Agents and Intelligent Robots

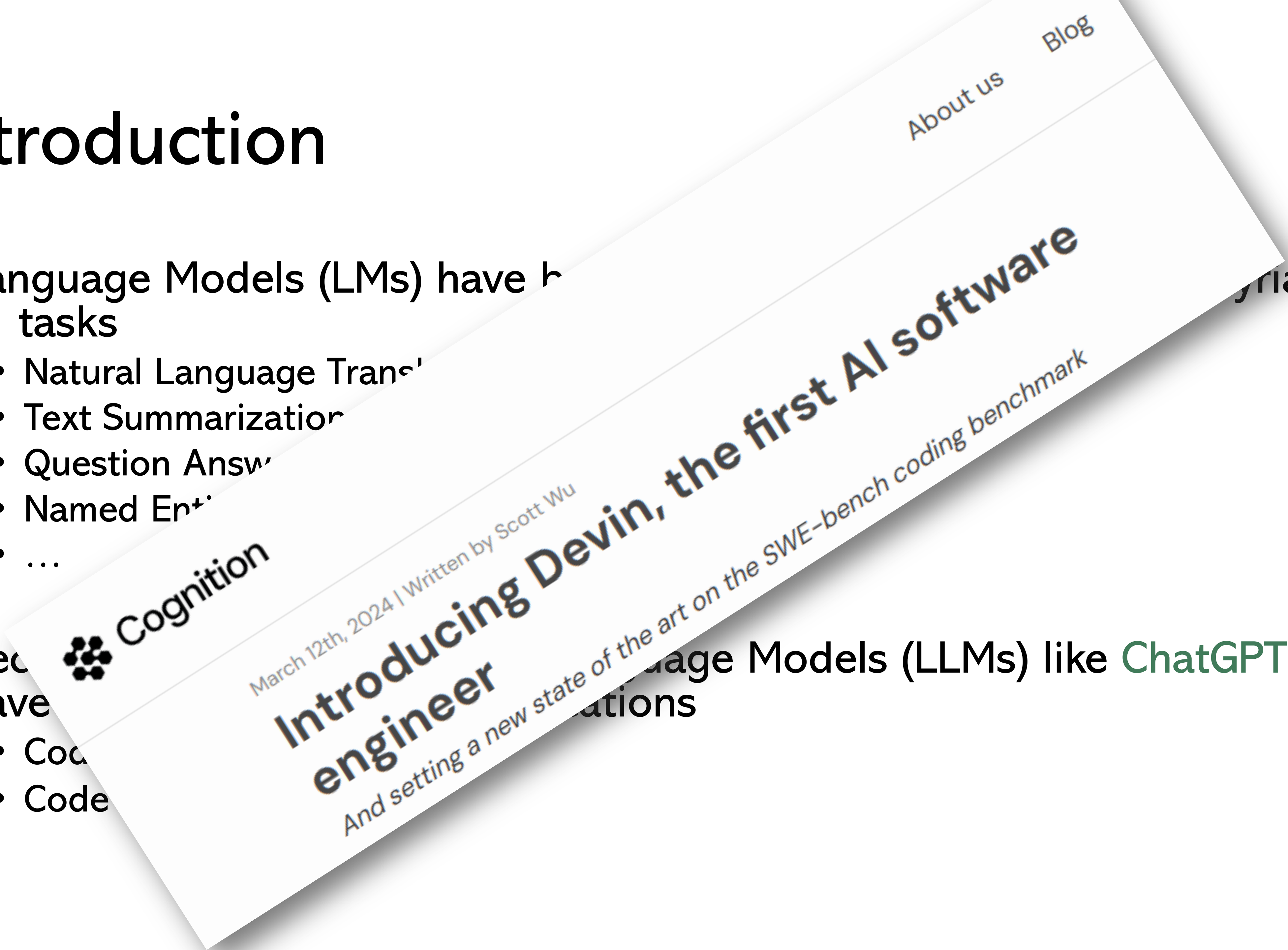
ASU Arizona State University

Introduction

- Language Models (LMs) have been shown to be successful in a myriad of tasks
  - Natural Language Translation
  - Text Summarization
  - Question Answering
  - Named Entity Recognition
  - …
- Recently, the advent of Large Language Models (LLMs) like ChatGPT have extended the list of applications
  - Code Generation
  - Code Explanation

Introduction

- Language Models (LMs) have b… …yriad of tasks
  - Natural Language Transl…
  - Text Summarizatio…
  - Question Answ…
  - Named Ent…
  - …
- Rec… …uage Models (LLMs) like ChatGPT have … …ations
  - Cod…
  - Code…



Motivation

- Increasing interest in using LLMs in system synthesis/explanation
  - Needs LLMs to generate/translate formal syntax (e.g. source code)

Q: Can we use LLMs for generating/interpreting formal syntax?

How do you create a scalable benchmark for testing LLMs?

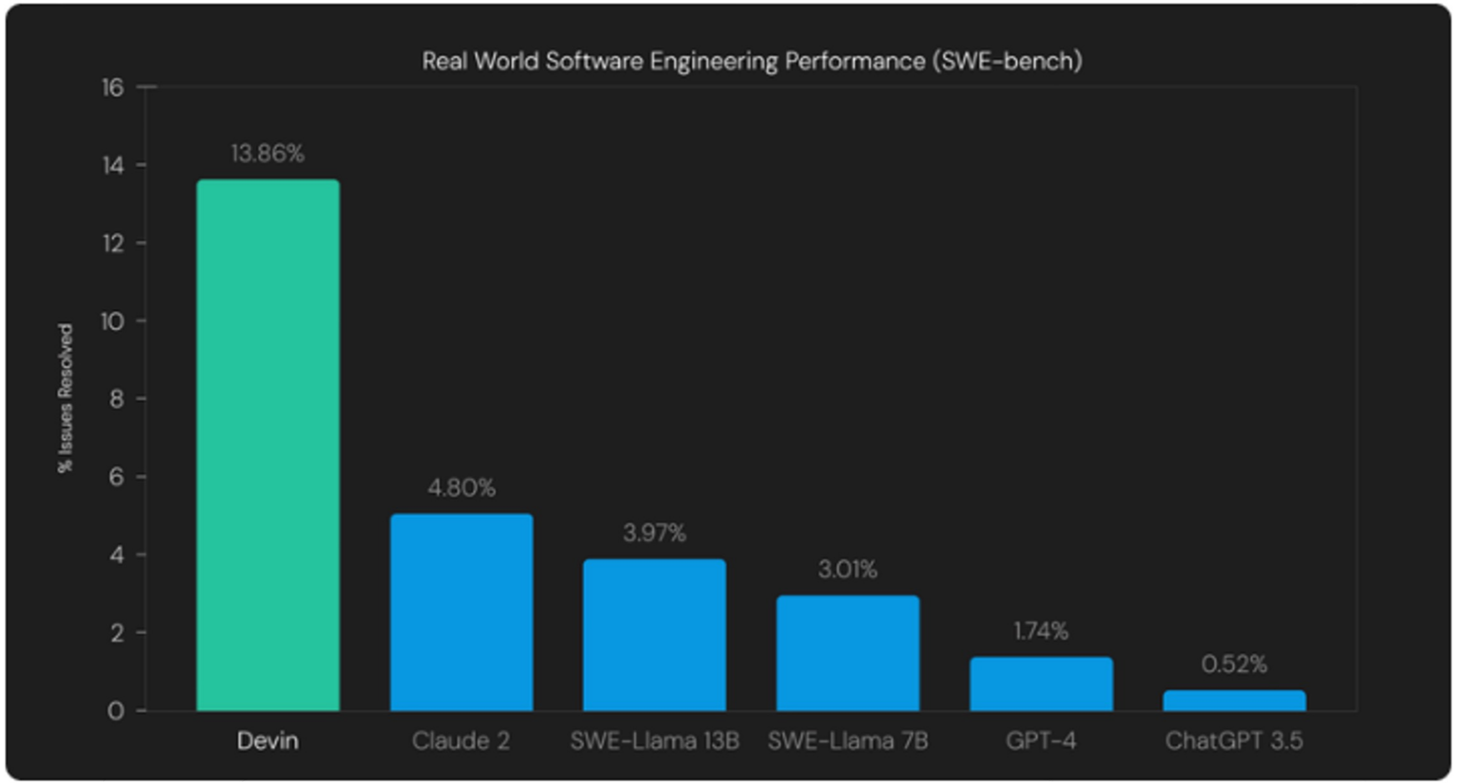
Key Challenges

1. What datasets to use?
   - Need datasets that are not already in the training set of LLMs
   - Need datasets that can be well-characterized (e.g. complexity of clauses)

2. How do we automatically evaluate LLMs?
   - Traditionally, humans annotate ground-truth data (very expensive)
   - Can we do this automatically?

Key Cl

1. What
    - Need … ls
    - Need … / of clauses)



*Devin was evaluated on a random 25% subset of the dataset. Devin was unassisted, whereas all other models were assisted (meaning the model was told exactly which files need to be edited).

2. How d
    - Tradit … nsive)
    - Can w

(Large) Language Models: (L)LMs

Given an input x consisting of a sequence of tokens, a (large) language model, $p_\theta(x)$, parameterized by θ, returns a sequence of tokens $y \sim p_\theta(y|x')$.

- The input x is known as a prompt (usually contains the input and question)
- The output y is conditioned on the context window size x'
  - Critical for performance (serves as short-term memory)

RU **You**

Summarize the following text using one sentence.

AI systems are increasingly interacting with users who are not experts in AI. This has led to growing calls for better safety assessment and regulation of AI systems. However, broad questions remain on the processes and technical approaches that would be required to conceptualize, express, manage, and enforce such regulations for adaptive AI systems, which by nature, are expected to exhibit different behaviors while adapting to evolving user requirements and deployment environments.

This symposim will foster research and development of new paradigms for assessment and design of AI systems that are not only efficient according to a task-based performance measure, but also safe to use by diverse groups of users and compliant with the relevant regulatory frameworks. It will highlight and engender research on new paradigms and algorithms for assessing AI systems' compliance with a variety of evolving safety and regulatory requirements, along with methods for expressing such requirements.

We also expect that the symposium will lead to a productive exchange of ideas across two highly active fields of research, viz., AI and formal methods. The organization team includes active researchers from both fields and our pool of invited speakers features prominent researchers from both areas.

Please feel free to send workshop related queries at: aia2024.symposium@gmail.com.

ChatGPT

The text discusses the increasing need for better safety assessment and regulation of AI systems due to their growing interaction with non-expert users, while proposing a symposium to promote research on new paradigms for designing AI systems that are efficient, safe, and compliant with regulatory frameworks, aiming to foster collaboration between AI and formal methods researchers.

Formal Syntax Translation: $NL \rightarrow SAT$

Given an input prompt NL in natural language, $NL \rightarrow SAT$ converts NL to formal syntax SAT s.t. SAT accurately represents NL

You
Convert the following sentence to first-order logic.
All men are mortal. Socrates is a man. Therefore, Socrates is mortal

ChatGPT
In first-order logic, we can represent the given sentence as follows:

Let $M(x)$ represent "x is mortal" and $H(x)$ represent "x is a man". Then, the sentence can be represented as:

$$\forall x(H(x) \rightarrow M(x)), \quad H(\text{Socrates}) \Rightarrow M(\text{Socrates})$$

R

Convert the following sentence to python code.
Function x takes two arguments a and b and returns the sum of a and b.

Gemini

Show drafts

Sure, here's the code:

Python

```python
def x(a, b): return a + b

# Example usage
print(x(5, 3))
```

Formal Syntax Interpretation: $SAT \rightarrow NL$

Given an input prompt SAT in formal syntax, $SAT \rightarrow NL$ converts SAT to natural language NL s.t. NL accurately represents SAT

RK Explain the following propositional formula:

(p1 and (p2 or (not p3))

Be concise in your response.

→ Claude →

The formula represents the conjunction of proposition p1 with a disjunction where either p2 is true or the negation of p3 is true. For the entire formula to be true, p1 must be true, and at least one of p2 or the negation of p3 must be true.

Copy Retry

RU **You**

Explain the following PDDL action concisely:

```
(:action grasp
  :parameters (?gripper - gripper ?object - object)
  :precondition (and (empty ?gripper))
  :effect (and (holding ?object) (not (empty ?gripper))))
```

→ →

ChatGPT

This PDDL action defines the "grasp" action, which takes a gripper and an object as parameters. It can only be executed if the gripper is empty, and upon execution, it makes the gripper hold the object while marking the gripper as not empty.

Our Contributions

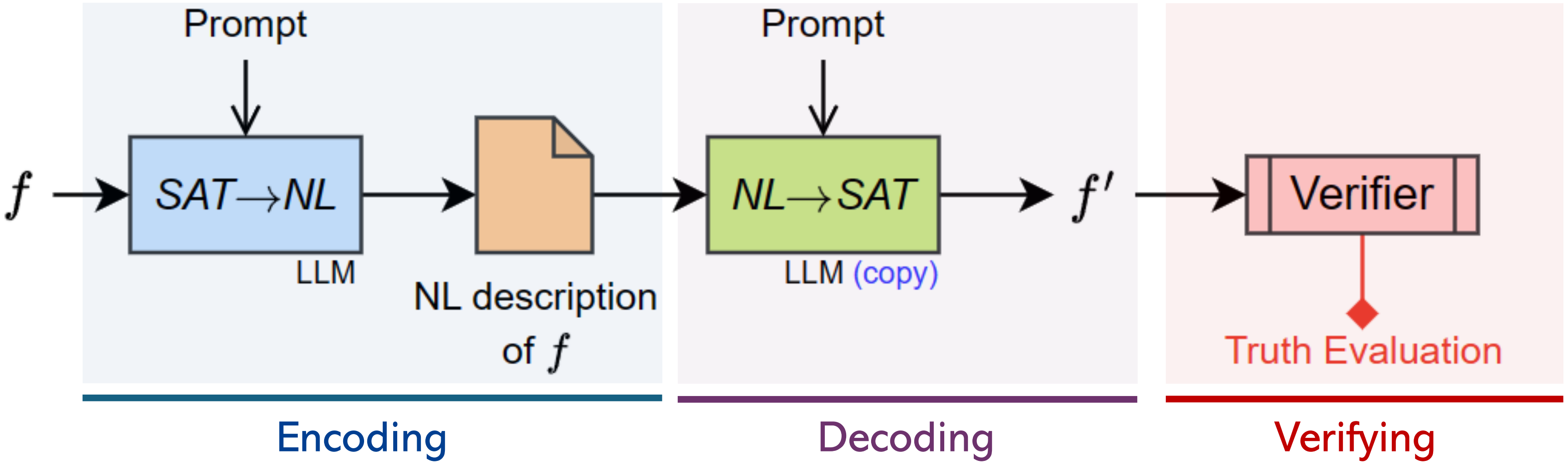
We present an approach for automatic evaluation of $NL \leftrightarrow SAT$

1. Automatic, ground-truth data generation
   - We generate data using the grammar of the formal language being tested
   - Formulae variables are perturbed to generate OOD data

2. Two LLM copies for automatic evaluation of $NL \leftrightarrow SAT$
   - We use an encoder-decoder scheme for automatic evaluation of $NL \leftrightarrow SAT$
   - No human input required in the process

3. Extensive empirical evaluation
   - We present an extensive evaluation using several SOTA LLMs

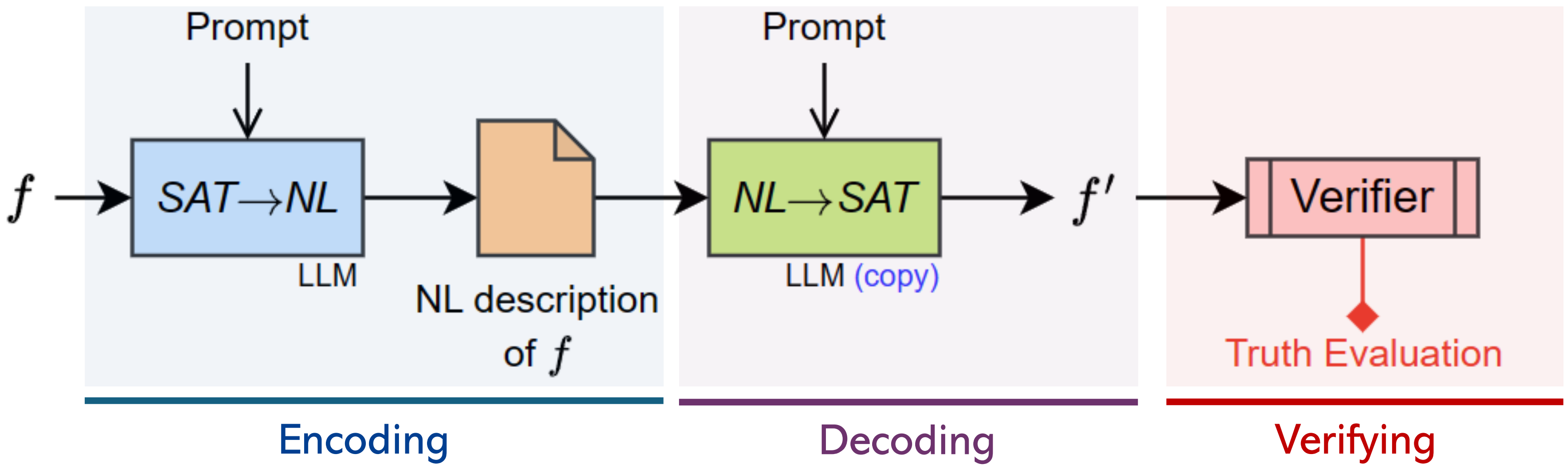
Automatic Assessment of $NL \leftrightarrow SAT$ using LLMs



Given a statement f in formal syntax, we

a) Encoding: Convert it to an NL description X using $SAT \rightarrow NL$ using an LLM
b) Decoding: We use a copy of the LLM to convert X to formal syntax f' using $NL \rightarrow SAT$
c) Verifying: We use a verifier (such as a theorem prover) to check if $f \equiv f'$

Example



$$(a \wedge b)$$

You

convert the following propositional logic formula to natural language without explicitly saying it.

(a and b)

ChatGPT

The combination of both a and b.

You

Convert the following text to propositional logic. Only state the formula as the answer.

The combination of both a and b.

ChatGPT

(a ∧ b)

Microsoft Research

Z3

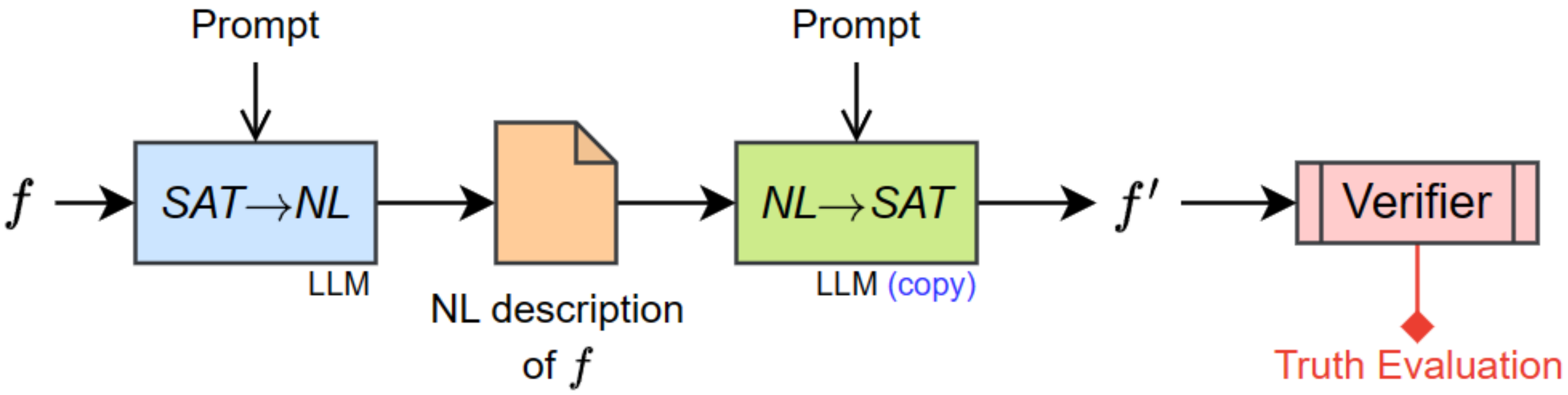
TRUE

Scalable Datasets

$$
\begin{aligned}
\mathcal{G}_{fol} : f &\rightarrow q, \ldots s|s \\
q &\rightarrow \forall\text{variable}|\exists\text{variable} \\
s &\rightarrow \text{predicate}|\text{predicate}(t, \ldots)|s \vee s|s \wedge s|\neg s \\
\text{predicate} &\rightarrow p_1|p_2| \ldots \\
t &\rightarrow \text{constant}|\text{variable} \\
\text{constant} &\rightarrow X_1|X_2| \ldots \\
\text{variable} &\rightarrow x_1|x_2| \ldots
\end{aligned}
$$

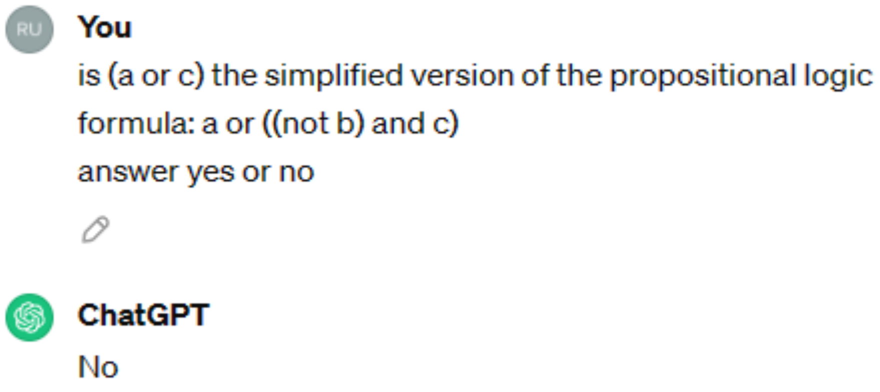
- Generated using language grammars
  - Can be scaled up easily (without requiring human annotation) as LLM capabilities evolve
- Can be stratified easily based on type of language produced
  - Example: k-SAT propositional logic formulae

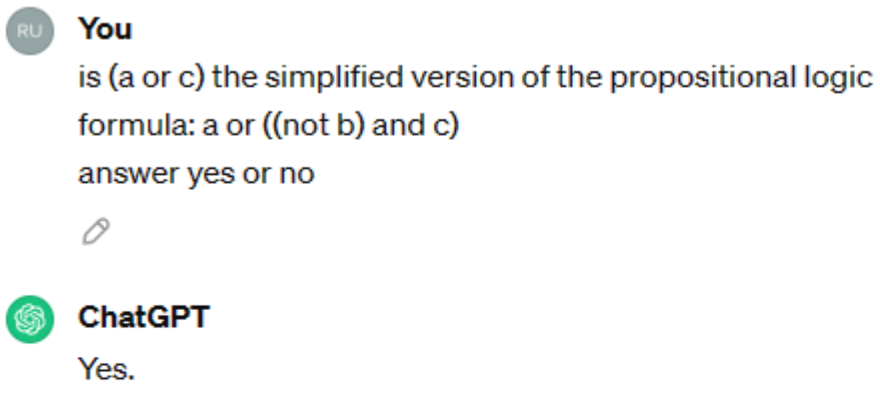
No/Minimal Human Input Needed



- Humans either not required at any point in the pipeline or trivially required
  - Example: Setting up parameters of the dataset generation etc.

Robust against Hallucinations





- Many techniques (in contrast to ours), use LLMs as a self-critique verifier
- We use well-defined reasoning processes (external verifier) for truth evaluation
  - This makes our approach less susceptible to hallucinations

Experiments

- We performed an empirical evaluation with two popular forms of formal syntax
    1. Propositional logic
    2. First-order logic

- We used several SOTA LLMs in our evaluation
    1. ChatGPT (GPT-3.5-turbo)
    2. GPT-4
    3. Gemini (Pro)
    4. Mistral (7B Instruct)

Our results show that LLMs are currently not suitable for deployment in formal specification-based tasks.

Propositional Logic

A statement or formula in propositional logic consists of a set of propositions connected together with logical connectives

- Grammar: $\mathcal{G}_{sat} : p \rightarrow x | p \vee p | p \wedge p | \neg p$

  where x is a proposition (e.g. $x_1 =$ All men are mortal)

  Example: $f = (x_1 \wedge x_2) \vee \neg x_3$

- We used Z3 to validate whether $f_1 \equiv f_2$ for two formulae: f_1, f_2

Prompt Engineering

- LLM performance quite often depends upon the prompt
- We ensured the efficacy of our prompts by testing different prompts on (k, m)-CNF (Canonical Normal Form) formulae:

(k, m)-CNF formulae are propositional logic formulae expressed as

$f = f_1 \wedge \ldots \wedge f_m$

where

$f_i = p_1 \vee \ldots \vee p_k$ with $p_i \in \{x_i, \neg x_i\}$ where x_i is a proposition from the set $\{x_1, \ldots, x_n\}$

- We tested different prompts on a dataset of 400 such formulae with $k = 3,\ n = 3$, and $m \in [3, 21]$ until at least one LLM had $\geq 95\%$ accuracy on $NL \leftrightarrow SAT$
  - For a given k, m, the formula has $(k-1)m$ operators.

Dataset Generation

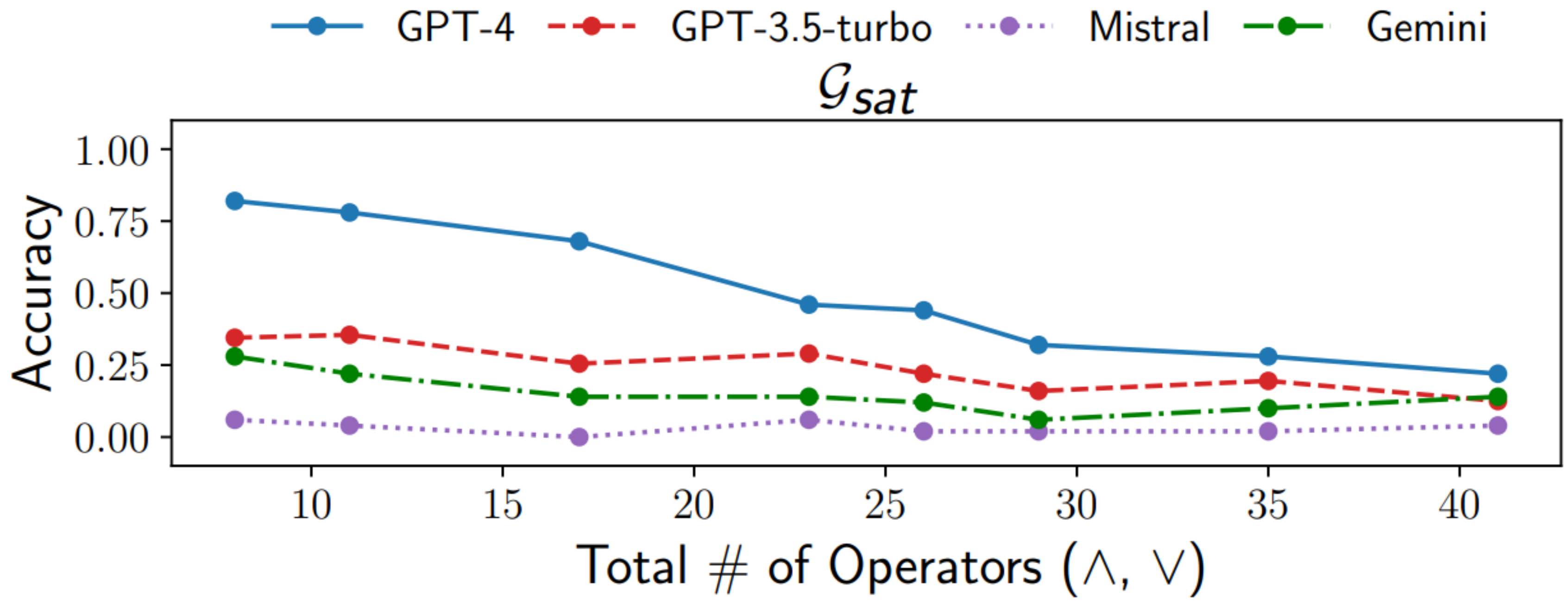
We used the following grammar for generating formulae:

$$\mathcal{G}_{sat} : p \rightarrow x | p \vee p | p \wedge p | \neg p$$

- Similar formula sizes as those in prompt engineering step
    - # of operators used as metric for generating formulae.

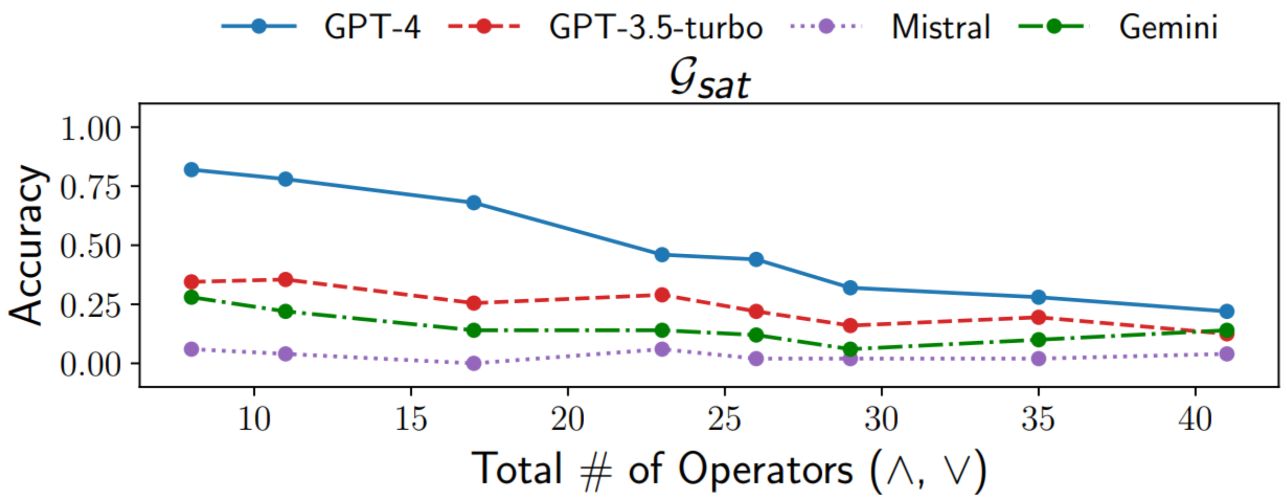
- We used Z3 as the verifier for testing logical equivalence

Results: (5 runs, higher values better)



- GPT-4 outperforms other LLMs
- LLMs cannot perform $NL \leftrightarrow SAT$ satisfactorily in propositional logic

Common Failures



$NL \rightarrow SAT$ failures

- One key failure was unable to capture the parentheses correctly
- Another was a bias from the LLMs knowledge based wherein it used symbols other than those in the prompt (e.g. $\rightarrow$)

$SAT \rightarrow NL$ failures

Assuming $SAT \rightarrow NL$ was correct and a human could correctly decode the NL expression

- LLMs like Gemini would often hallucinate during this phase

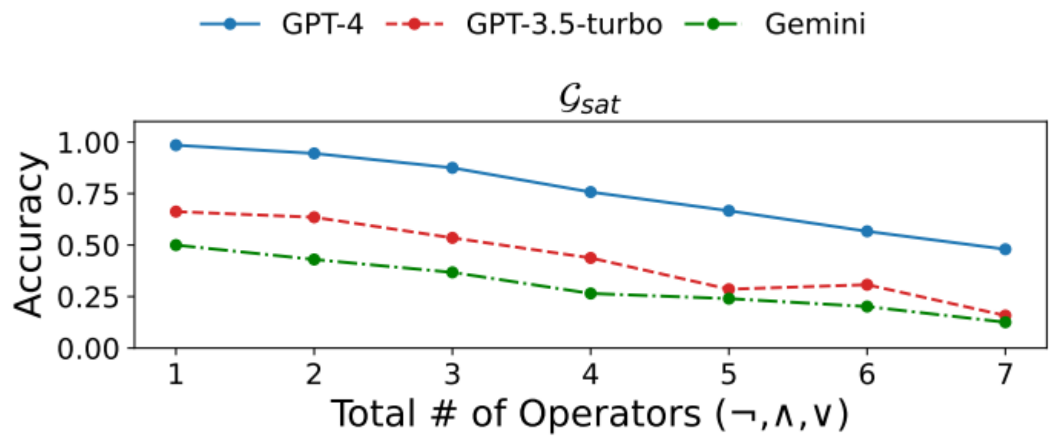
First-order Logic (FOL)

A statement or formula in first-order logic consists of a set of predicates connected together with logical connectives and quantifiers

$$
\begin{aligned}
\mathcal{G}_{fol}: f &\rightarrow q, \ldots s|s \\
q &\rightarrow \forall \text{variable}|\exists \text{variable} \\
s &\rightarrow \text{predicate}|\text{predicate}(t, \ldots)|s \vee s|s \wedge s|\neg s \\
\text{predicate} &\rightarrow p_1|p_2| \ldots \\
t &\rightarrow \text{constant}|\text{variable} \\
\text{constant} &\rightarrow X_1|X_2| \ldots \\
\text{variable} &\rightarrow x_1|x_2| \ldots
\end{aligned}
$$

We used a similar prompt engineering step for FOL.

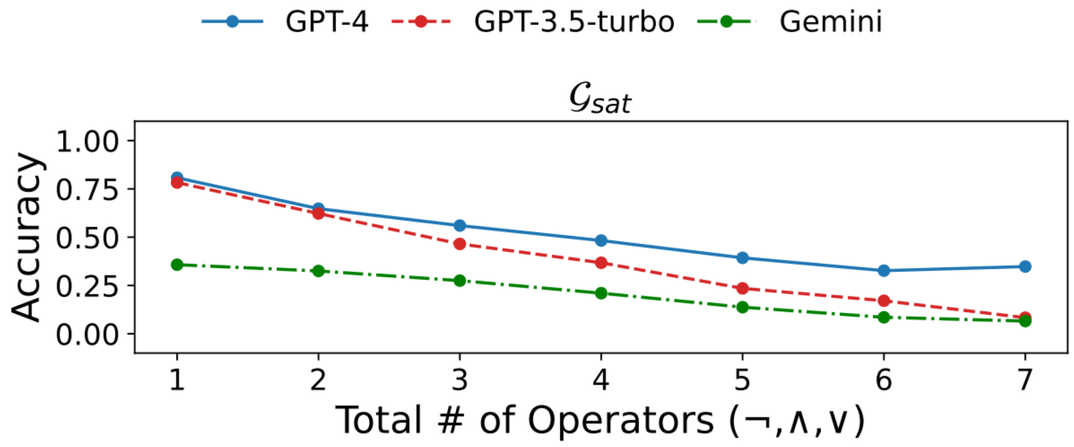
Results: Randomized predicates



- Results with randomized versions of predicates
  - Eg: p1(o1) and p2(o1, o4)
- Performance of LLMs is about 50% for small formulae.

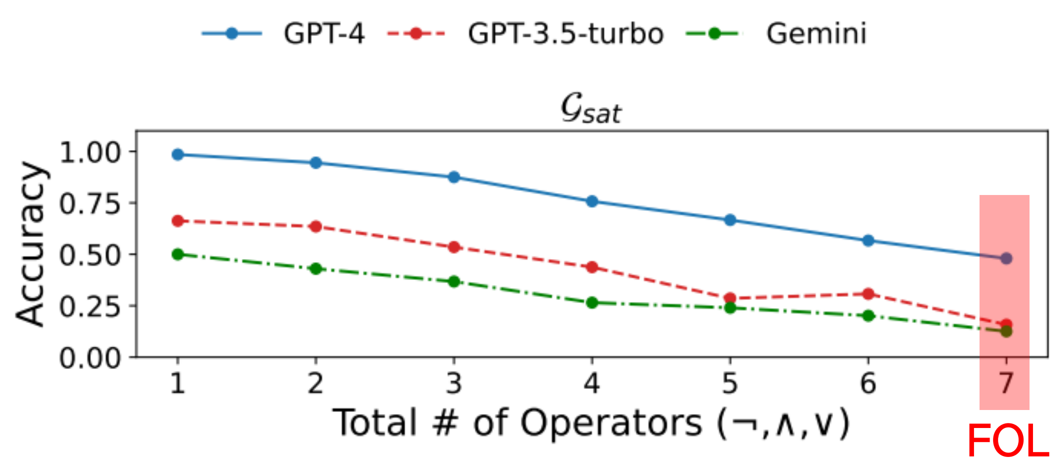
Results: English language-based predicates

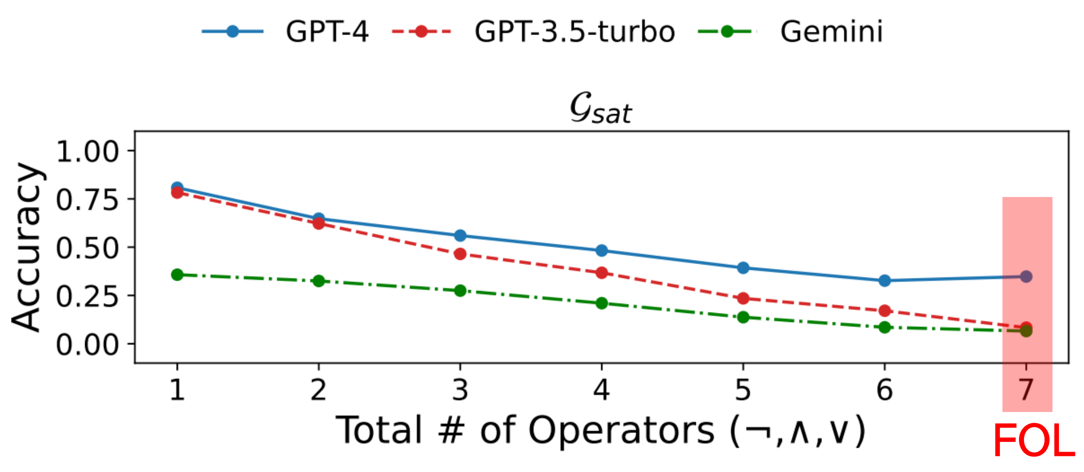
$$
\begin{aligned}
\mathcal{G}_{fol} : f &\rightarrow q, \ldots s | s \\
q &\rightarrow \forall \text{variable} | \exists \text{variable} \\
s &\rightarrow \text{predicate} | \text{predicate}(t, \ldots) | s \vee s | s \wedge s | \neg s \\
\text{predicate} &\rightarrow \text{Friends} | \text{Enemies} | \ldots \\
t &\rightarrow \text{constant} | \text{variable} \\
\text{constant} &\rightarrow \text{John} | \text{William} | \ldots \\
\text{variable} &\rightarrow x_1 | x_2 | \ldots
\end{aligned}
$$



- Results with English language-based versions of predicates
  - Predicates were pulled from a vocabulary
- Slightly worse performance than randomized predicates

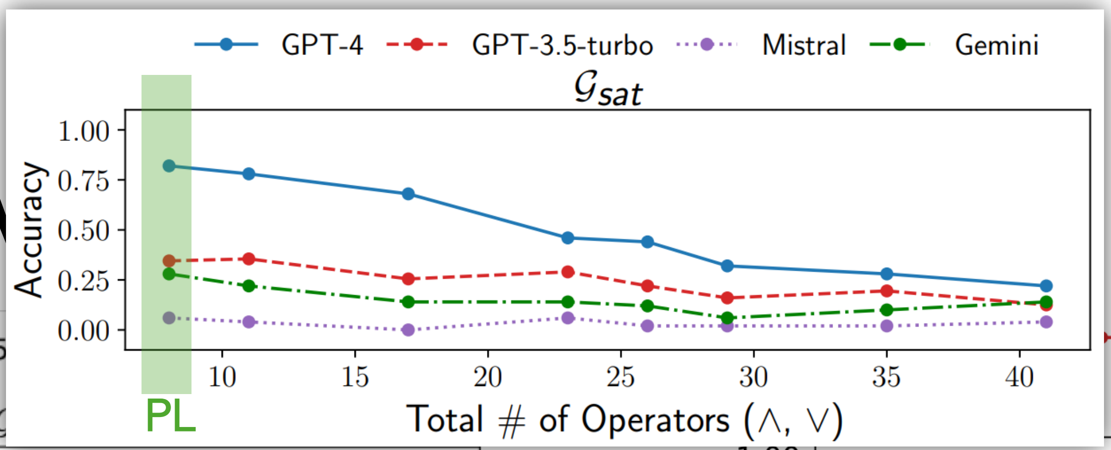
Results: Overall

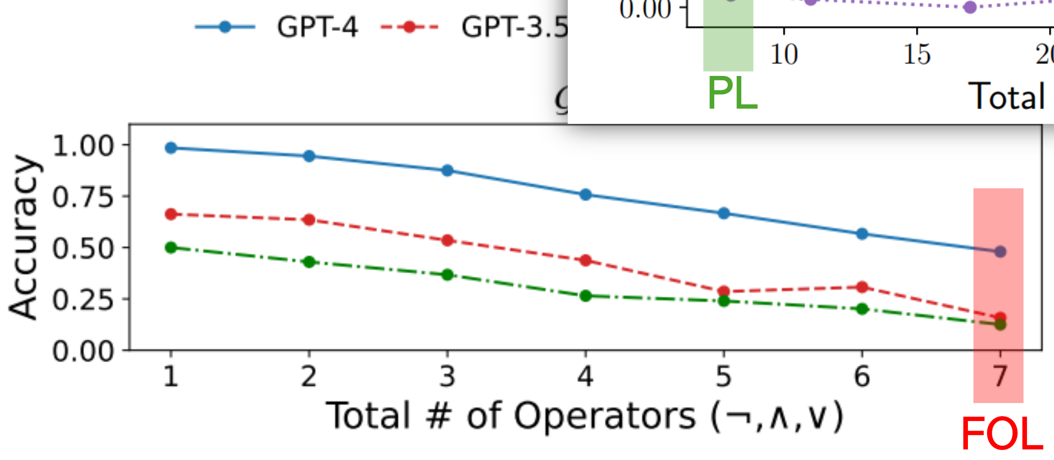


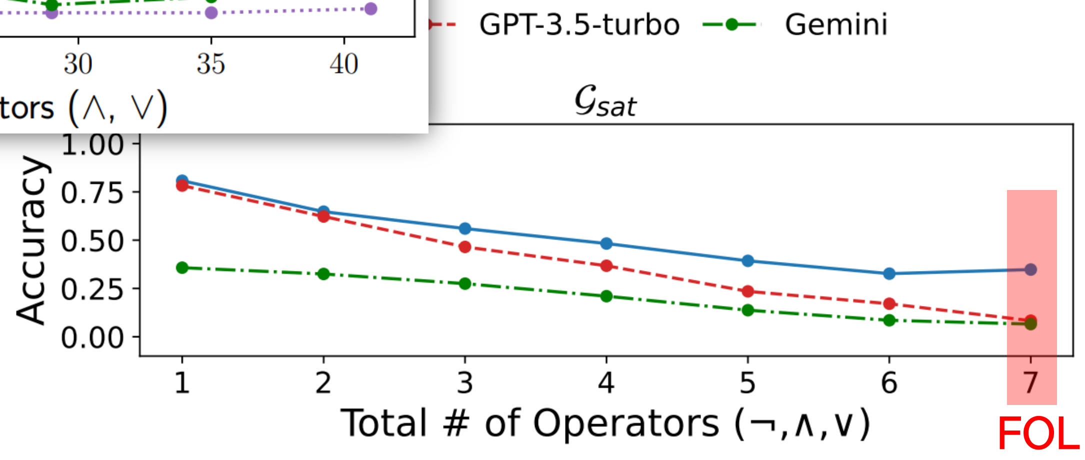


- Our results show that LLMs cannot translate satisfactorily even for FOL
- Much worse than propositional logic (PL)
  - Results much worse, much earlier: at 7 #ops, accuracy=75% (50%) for PL (FOL)
  - In PL, we did not even count negations as operators!

Results: Ov







- Our results show that LLMs cannot translate satisfactorily even for FOL
- Much worse than propositional logic (PL)
  - Results much worse, much earlier: at 7 #ops, accuracy=75% (50%) for PL (FOL)
  - In PL, we did not even count negations as operators!

Conclusions

- We introduced an effective technique for assessing LLMs in the sphere of formal syntax
- Our approach is automatic and is developed on well-defined reasoning processes making it less susceptible to self-evaluation/critic based hallucinations

Future Work

- Expand coverage to different kinds of formal syntax: (e.g. PDDL)