

LLMs and Software Engineering

History, Landscape, and Outlook

Who am I?

Shin Yoo

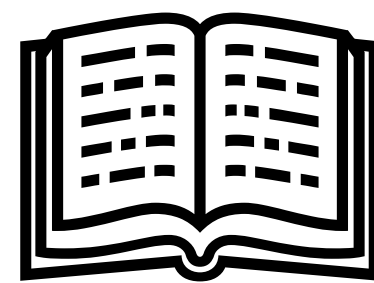
- Associate Prof@KAIST
- Leads Computational Intelligence for Software Engineering Group (<https://coinse.github.io>)
- Search Based Software Engineering, Automated Debugging, Software Testing...



“GPT-3 was introduced in 2020 - surely it is too early to speak of history...?”

LLMs and Software Engineering

History, Landscape, and Outlook



History

Let's go back to 2012

Hindle et al., ICSE 2012

- One of my favourite papers: On Naturalness of Software (<https://dl.acm.org/doi/10.5555/2337223.2337322>)
- “Programming languages, in theory, are complex, flexible and powerful, but the programs that real people actually write are mostly simple and rather repetitive, and thus they have usefully predictable statistical properties that can be captured in statistical language models and leveraged for software engineering tasks.”
- But what is “naturalness”?

What is “natural” about language?

- Natural language refers to ordinary languages that occur naturally in human community “by process of use, repetition, and change without conscious planning or premeditation” ([Wikipedia](#))
- From the statistical point of view, it means that most of our utterances are simple, repetitive, and therefore predictable.
 - Surely this is how we all learn language.



John: Hi, nice to meet you. How are you?

Mary: I'm _____, _____ _____. _____ _____?

a) fine, thank you. And you?

b) okay, I guess. But why?

What about code?

- It is not “natural”, in the sense that we have artificially created the grammar for programming languages.
- Programming languages do evolve, but how?
 - Intentionally? New grammars, language consortiums, etc...
 - Gradually? Languages do affect each other, a newer and more popular style eventually gets accepted, etc...

Python: for _ _ _ _ _ ...

a) i in range

b) (int i = 0;

Java: for _ _ _ _ _ ...

a) i in range

b) (int i = 0;

Statistical Language Model

- Given a set of tokens, \mathcal{T} , a set of possible utterances, \mathcal{T}^* , and a set of actual utterances, $\mathcal{S} \subset \mathcal{T}^*$, a language model is a probability distribution p over utterances $s \in \mathcal{S}$, i.e., $\forall s \in \mathcal{S} [0 < p(s) < 1 \wedge \sum_{s \in \mathcal{S}} p(s) = 1$
- An utterance (or a sentence) is a sequence of tokens (or words). Suppose we have N tokens, a_1, a_2, \dots, a_N that consist s . What is $p(s)$?
 - $p(s) = p(a_1)p(a_2 | a_1)p(a_3 | a_1, a_2)p(a_4 | a_1, a_2, a_3) \dots p(a_N | a_1 \dots a_{N-1})$
 - But these conditional probabilities are hard to calculate: the only feasible approach would be count each utterance that qualifies, but \mathcal{S} is too big, let alone \mathcal{T}^* .

N-Grams

- Assumes Markov property, i.e., the next token is influenced only by those came immediately before (say, within the window of n tokens)!

- $p(a_i | a_1 \dots a_{i-1}) \simeq p(a_i | a_{i-3} a_{i-2} a_{i-1})$

- This is now much more tractable:

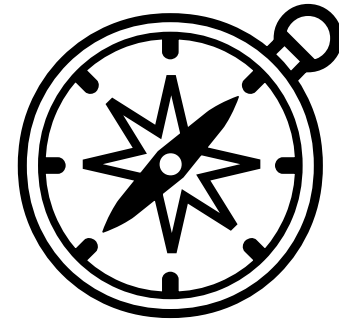
- $$p(a_i | a_{i-3} a_{i-2} a_{i-1}) = \frac{\text{count}(a_{i-3}, a_{i-2}, a_{i-1}, a_i)}{\text{count}(a_{i-3}, a_{i-2}, a_{i-1}, *)}$$

- Given some context, we can now compute the probability of the candidate token that comes next. In other words, we can predict the next token!

Large Language Model

(really, a very large statistical language model)

- Mainly Transformer-based DNNs that are trained to be an auto-regressive language model, i.e., given a sequence of tokens, it repeatedly tries to predict the next token.
- The biggest hype in SE research right now with an **explosive** growth, because:
 - They **seem to** get the semantics of the code and **work across natural and programming language**
 - **Emergent behavior** leading to very attractive properties such as in-context learning, Chain-of-Thoughts, or PAL



Landscape

Survey of the Explosion

ICSE 2023 Future of SE Track (<https://arxiv.org/abs/2310.03533>)

Large Language Models for Software Engineering: Survey and Open Problems

Angela Fan
Generative AI Team
Meta Platforms Inc.
New York, NY, USA

Beliz Gokkaya
PyTorch Team
Meta Platforms Inc.
Menlo Park, CA, USA

Mark Harman
Instagram Product Foundation
Meta Platforms Inc.
London, UK

Mitya Lyubarskiy
Developer Infrastructure
Meta Platforms Inc.
London, UK

Shubho Sengupta
FAIR
Meta Platforms Inc.
Menlo Park, CA, USA

Shin Yoo
School of Computing
KAIST
Daejeon, Korea

Jie M. Zhang
Department of Informatics
King's College London
London, UK

Abstract—This paper provides a survey of the emerging area of Large Language Models (LLMs) for Software Engineering (SE). It also sets out open research challenges for the application of LLMs to technical problems faced by software engineers. LLMs' emergent properties bring novelty and creativity with applications right across the spectrum of Software Engineering activities including coding, design, requirements, repair, refactoring, performance improvement, documentation and analytics. However, these very same emergent properties also pose significant technical challenges; we need techniques that can reliably weed out incorrect solutions, such as hallucinations. Our survey reveals the pivotal role that hybrid techniques (traditional SE

In particular, we are already able to discern important connections to (and resonance with) existing trends and well-established approaches and subdisciplines within Software Engineering. Furthermore, although we find considerable grounds for optimism, there remain important technical challenges, which are likely to inform the research agenda for several years. Many authors have highlighted, both scientifically and anecdotally, that hallucination is a pervasive problem for LLMs [1] and also that it poses specific problems for LLM-based SE [2]. As with human intelligence, hallucination means

[cs.SE] 11 Nov 2023

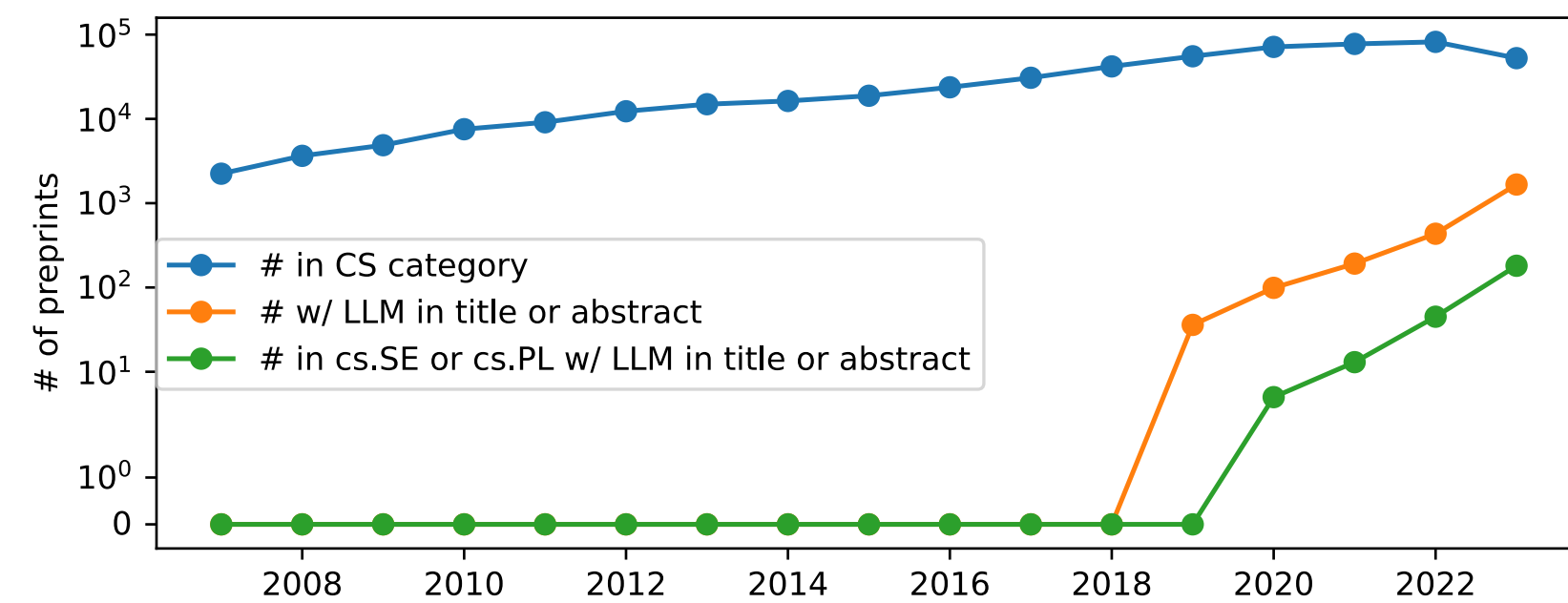


Fig. 2. Trends in number of arXiv preprints. The blue line denotes the number of preprints categorised under “CS”. The orange line denotes the number of preprints in AI (cs.AI), Machine Learning (cs.LG), Neural and Evolutionary Computing (cs.NE), Software Engineering (cs.SE), and Programming Language (cs.PL) whose title or abstract contains either “Large Language Model”, “LLM”, or “GPT”. The green line denotes the number of preprints in SE and PL categories whose title or abstract contains either “Large Language Model”, “LLM”, or “GPT”

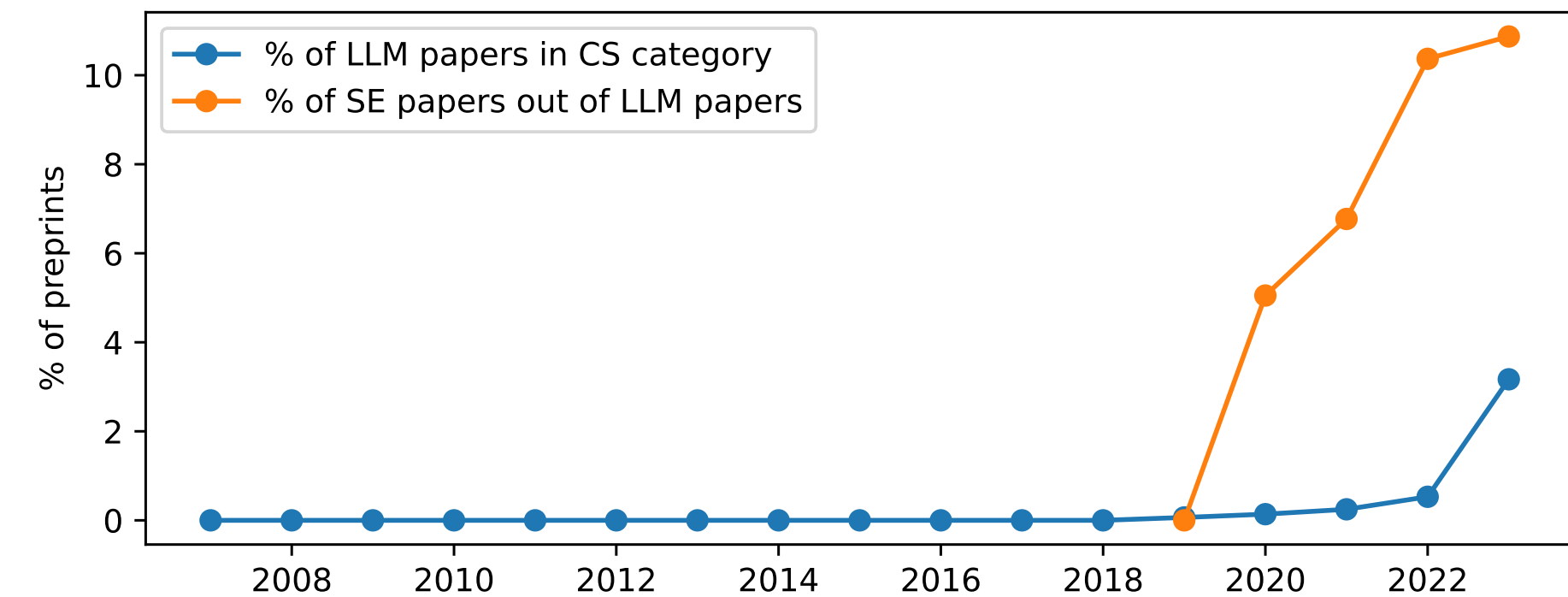


Fig. 3. Proportions of LLM papers and SE papers about LLMs. By “about LLMs”, we mean that either the title or the abstract of a preprint contains “LLM”, “Large Language Model”, or “GPT”. The blue line denotes the percentage of the number of preprints about LLMs out of the number of all preprints in the CS category. The orange line denotes the percentage of the number of preprints about LLMs in cs.SE and cs.PL categories out of all preprints about LLMs

<https://arxiv.org/abs/2310.03533>

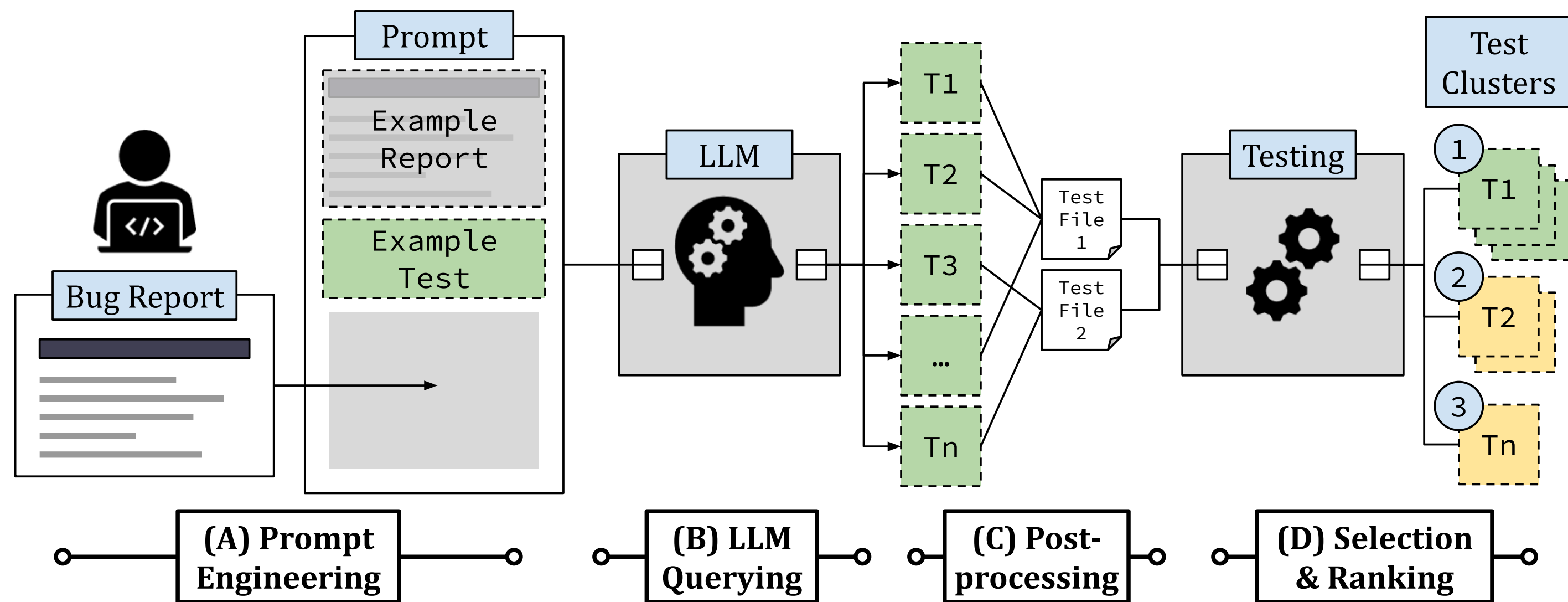
Does it engage with semantic across NL and PL?

An example: bug reproduction

- A classic challenge in automated testing: finding an input that executes specific branch in the code is easy, **deciding whether that execution was buggy or not is not easy (=oracle problem).**
- Bug reproduction is the task of reproducing a buggy execution based on bug report written in natural language.
 - Traditionally, the **only** “buggy behaviour” that can be automatically confirmed has been **crashes**.

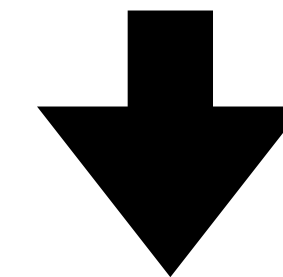
LLMs can make that connection.

LLM-based Bug Reproduction (Kang, Yoon & Yoo, ICSE 2023)



Title `assertContainsIgnoringCase` fails to compare `i` and `I` in `tr_TR` locale

See `org.assertj.core.internal.Strings#assertContainsIgnoringCase` [url]
I would suggest adding [url] verification to just ban `toLowerCase()`, `toUpperCase()` and other unsafe methods: #2664



```
public void testIssue952() {  
    Locale locale = new Locale("tr", "TR");  
    Locale.setDefault(locale);  
    assertThat("I").as("Checking in tr_TR locale")  
        .containsIgnoringCase("i");  
}
```



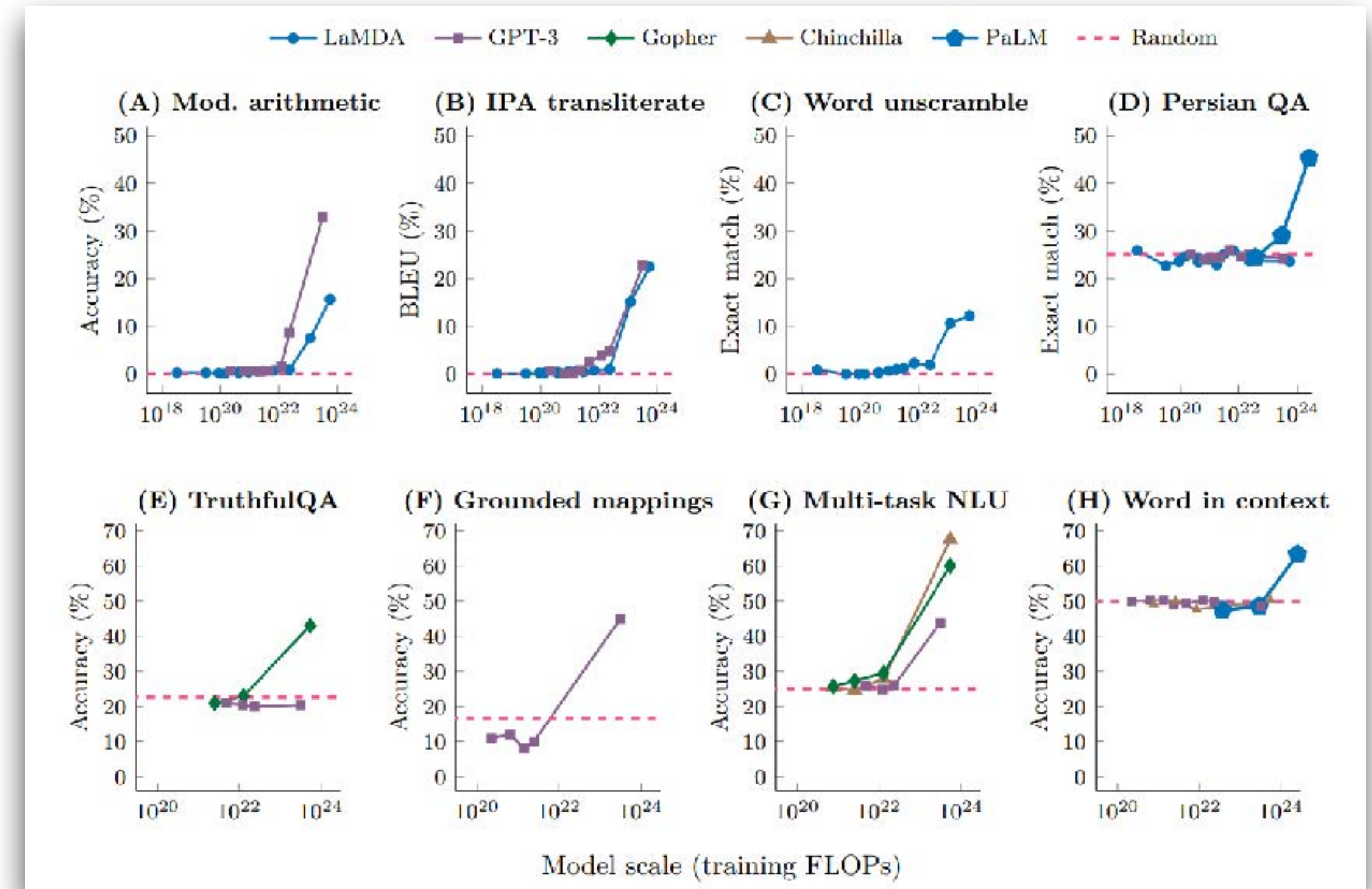
Sungmin Kang
(PhD Candidate)



Juyeon Yoon
(PhD Candidate)

What is an Emergent Behavior?

- Above certain size, LLMs change their behavior in interesting ways
- The point of change in slope is referred to as “breaks”



Caballero et al., <https://arxiv.org/abs/2210.14891>

In-context Learning

- Previously, getting a model for a specific task involved either dedicated model + training, or at least general pre-trained model + fine-tuning
- Above certain size, LLMs show the ability to perform in-context learning, i.e., they learn as part of their context (i.e., preceding tokens), leading to **prompt engineering**:
 - Few-shot learning: the context explains the problem, and gives a few examples of question-answer. LLMs can now answer an un-seen question.
 - Zero-shot learning: the context explains the problem as well as how it can be solved. LLMs can now answer an un-seen problem.

Chain-of-Thoughts

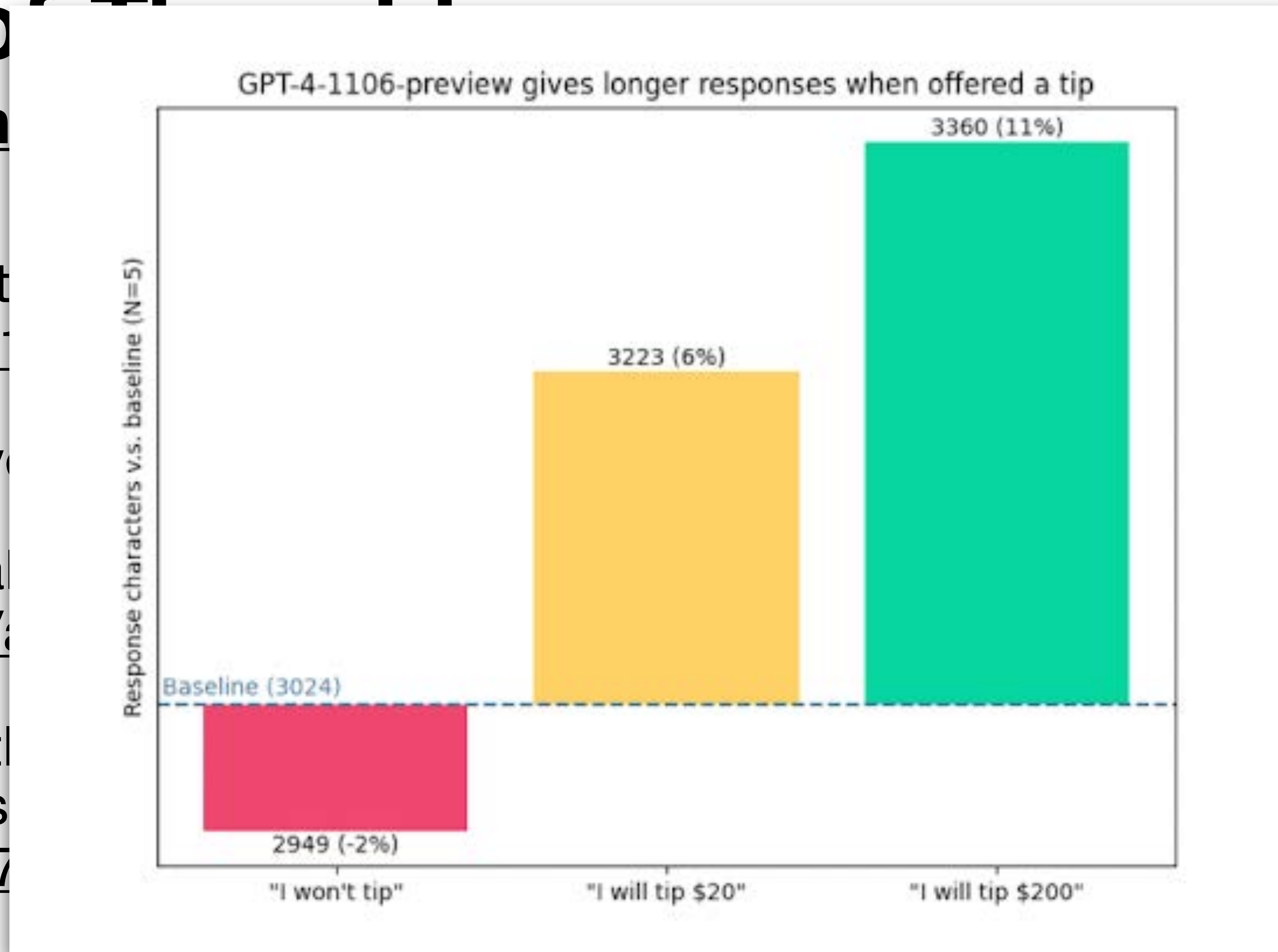
Wei et al., <https://arxiv.org/abs/2201.11903>

- Underneath, LLMs are doing autocompletion, not any other type of reasoning: they appear to be capable of rational inference because the corpus they are trained include traces of logical reasoning.
- So, **conditioning** the model (with the context) to be more precise about the reasoning steps can result in generation of more accurate reasoning steps.
- Add “Let’s think in step by step” at the end of every prompt (<https://arxiv.org/abs/2205.11916>) 🤔 😐 😊

Chain-of

Wei et al., [https://arxiv.org/abs/2205.11171](#)

- Add “Let’s tip” to the prompt
- We have even found that the model produces longer responses when offered a tip
- If you make the tip amount larger, the model produces even longer responses
- Apparently, the model produces longer responses when offered a larger tip



<https://arxiv.org/>

[s \(https://](https://)

ge tip
google/status/



Program-Aided Language Models (PAL)

Gao et al., ICML 2023 (<https://arxiv.org/abs/2211.10435>)

- What is even more logical and step by step than natural language? Programming language :)
- Providing few-shot examples that are mixtures of NL and LP can enhance the reasoning capabilities of LLM

Program-aided Language models (this work)

Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 tennis balls.

```
tennis_balls = 5
```

```
2 cans of 3 tennis balls each is
```

```
bought_balls = 2 * 3
```

```
tennis_balls. The answer is
```

```
answer = tennis_balls + bought_balls
```

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

Model Output

A: The bakers started with 200 loaves

```
loaves_baked = 200
```

```
They sold 93 in the morning and 39 in the afternoon
```

```
loaves_sold_morning = 93
```

```
loaves_sold_afternoon = 39
```

```
The grocery store returned 6 loaves.
```

```
loaves_returned = 6
```

```
The answer is
```

```
answer = loaves_baked - loaves_sold_morning
```

```
- loaves_sold_afternoon + loaves_returned
```

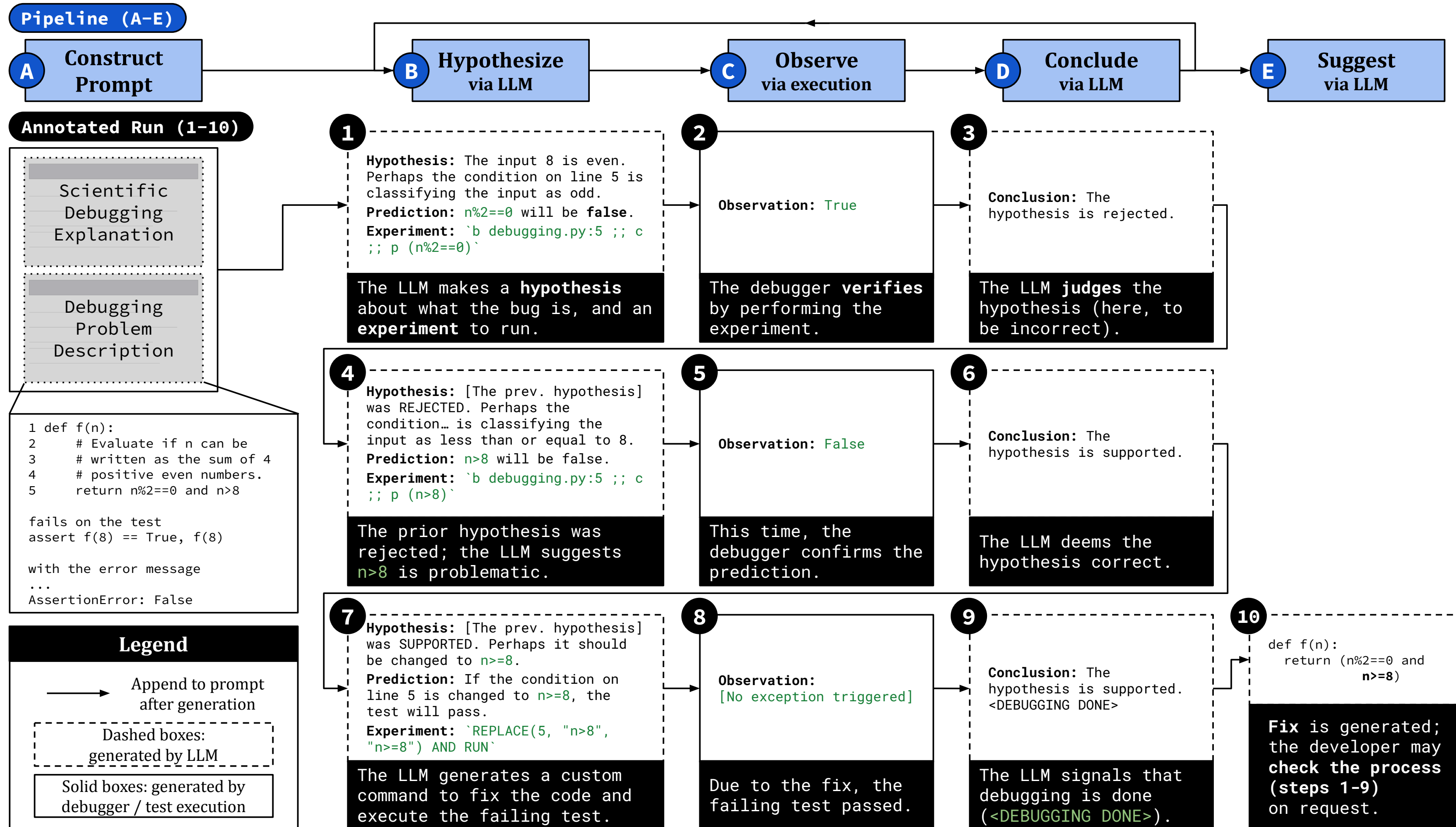
```
>>> print(answer)
```

```
74
```



Zero-shot Automated Debugging

Kang et al., <https://arxiv.org/abs/2304.02195>



Sungmin Kang
(PhD Candidate)

ReAct

Yao et al., ICLR 2023 (<https://arxiv.org/abs/2210.03629>)

- What if we need external information for the in-context learning? In other words, can LLMs be given tools?
- Remember that this is still autocompletion:
 - LLMs can be taught to signal the need to invoke tools
 - Whenever LLMs need a tool invocation, we can do it ourselves and paste the outcome back into the context

```
(1d) ReAct (Reason + Act)

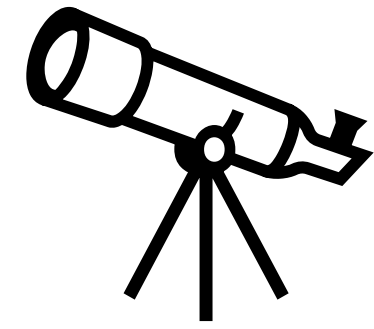
Thought 1: I need to search Apple Remote and find the
program it was originally designed to interact with.
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control introduced in
October 2005 by Apple ... originally designed to control the
Front Row media center program ...

Thought 2: Apple Remote was originally designed to control
the Front Row media center program. I need to search Front
Row next and find what other device can control it.
Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: ['Front Row
Seat to Earth', 'Front Row Motorsports', 'Front Row
(software)', ...]

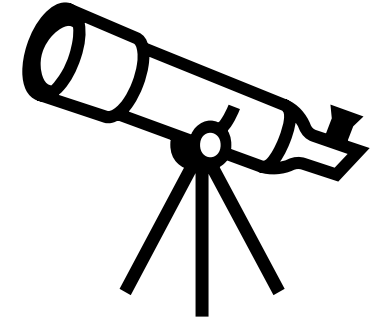
Thought 3: Front Row is not found. I need to search Front
Row (software) .
Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software ...

Thought 4: Front Row (software) is controlled by an Apple
Remote or the keyboard function keys. So the answer is
keyboard function keys.
Act 4: Finish[keyboard function keys]
```

ReAct: Synergizing Reasoning and Acting in Language Models, Yao et al., ICLR 2023
<https://arxiv.org/abs/2210.03629>



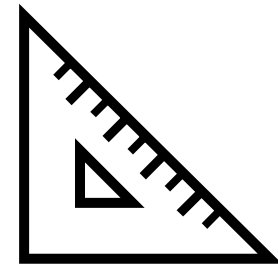
Outlook



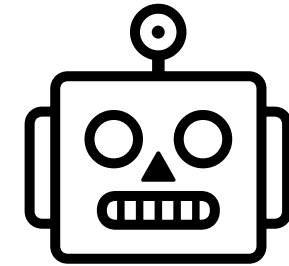
Outlook



Hallucination



Context Length



Agency

Hallucination

- LLM = (Statistical) Autocompletion = completion **not because it is the right choice, but because it is the most likely choice.**
- How do we filter out hallucinations?



We are still in the Chinese room

John Searle, “Mind, Brains, and Programs” in 1980

- Suppose we have a computer program that behaves as if it understands Chinese language.
- You are in a closed room with the AI program source code.
- Someone passes a paper with Chinese characters written on it, into the room.
- You use the source code as instruction to generate the response to the input, and sends the response out of the room.
- Do you understand Chinese language, or not?



Self-Consistency

Wang et al., ICLR 2023 (<https://arxiv.org/abs/2203.11171>)

- When sampling answers from an LLM, take multiple answers with high temperature.
- If there is an answer that has the majority among the sampled answers, it is more likely to be the correct one.

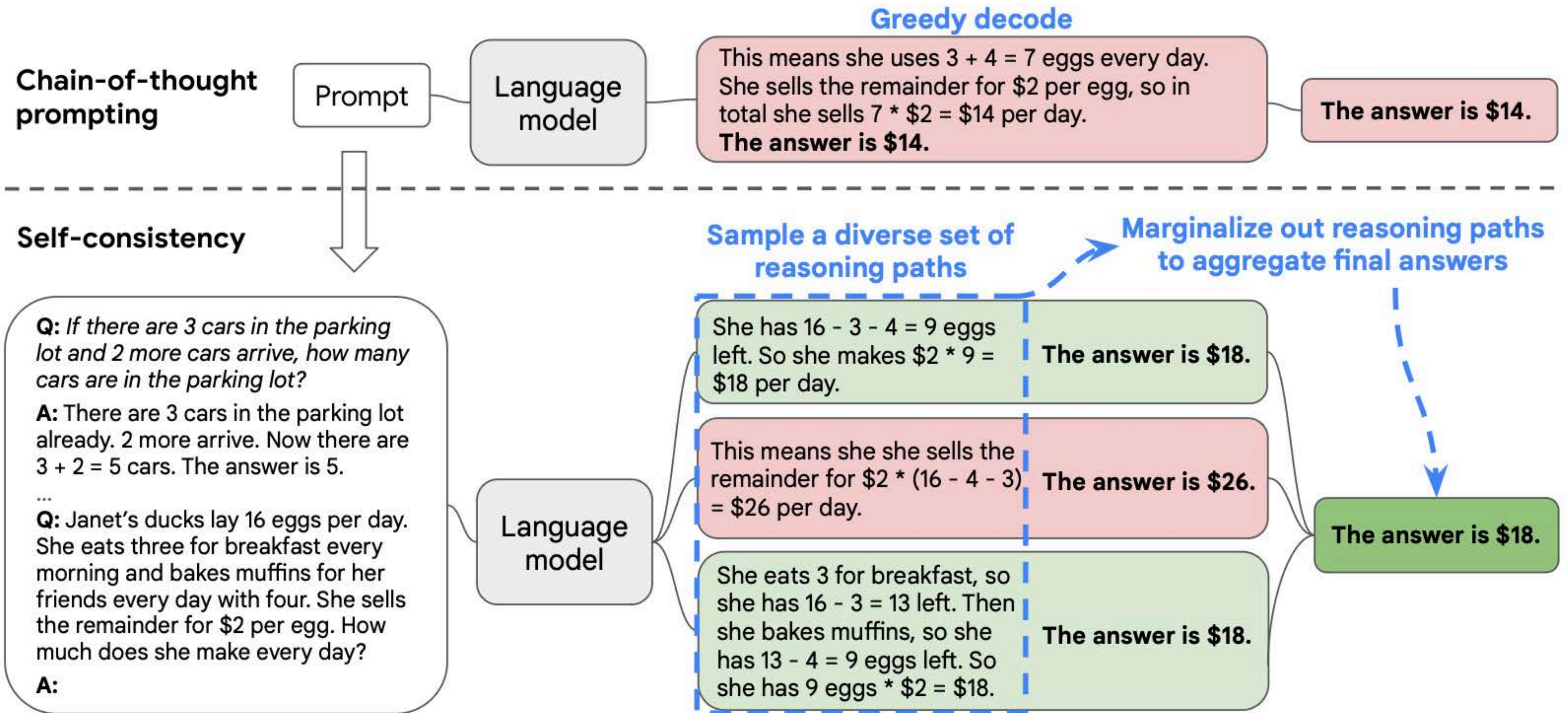
Published as a conference paper at ICLR 2023

SELF-CONSISTENCY IMPROVES CHAIN OF THOUGHT REASONING IN LANGUAGE MODELS

Xuezhi Wang^{†‡} Jason Wei[†] Dale Schuurmans[†] Quoc Le[†] Ed H. Chi[†]
Sharan Narang[†] Aakanksha Chowdhery[†] Denny Zhou^{†§}
[†]Google Research, Brain Team
[‡]xuezhiw@google.com, [§]dennyzhou@google.com

ABSTRACT

Chain-of-thought prompting combined with pre-trained large language models has achieved encouraging results on complex reasoning tasks. In this paper, we propose a new decoding strategy, *self-consistency*, to replace the naive greedy decoding used in chain-of-thought prompting. It first samples a diverse set of reasoning paths instead of only taking the greedy one, and then selects the most consistent answer by marginalizing out the sampled reasoning paths. Self-consistency leverages the intuition that a complex reasoning problem typically admits multiple different ways of thinking leading to its unique correct answer. Our extensive empirical evaluation shows that self-consistency boosts the performance of chain-of-thought prompting with a striking margin on a range of popular arithmetic and commonsense reasoning benchmarks, including GSM8K (+17.9%), SVAMP (+11.0%), AQuA (+12.2%), StrategyQA (+6.4%) and ARC-challenge (+3.9%).



But... really? That simple...?



|| 123456789 | 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

“the face of a man who is surprised that the answer was so simple.”

LLM-Based Bug Reproduction

Kang et al., ICSE 2023

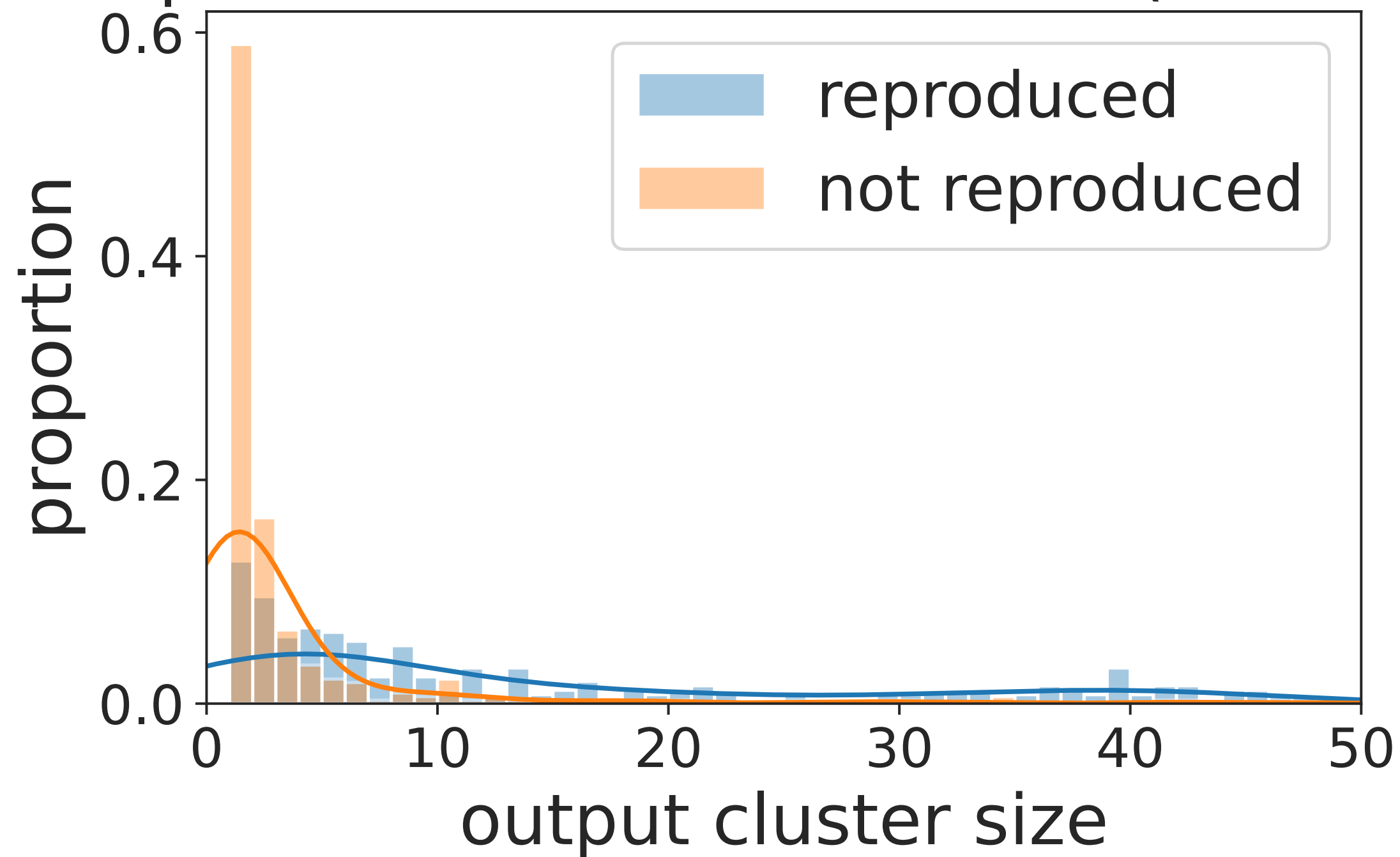


Sungmin Kang
(PhD Candidate)

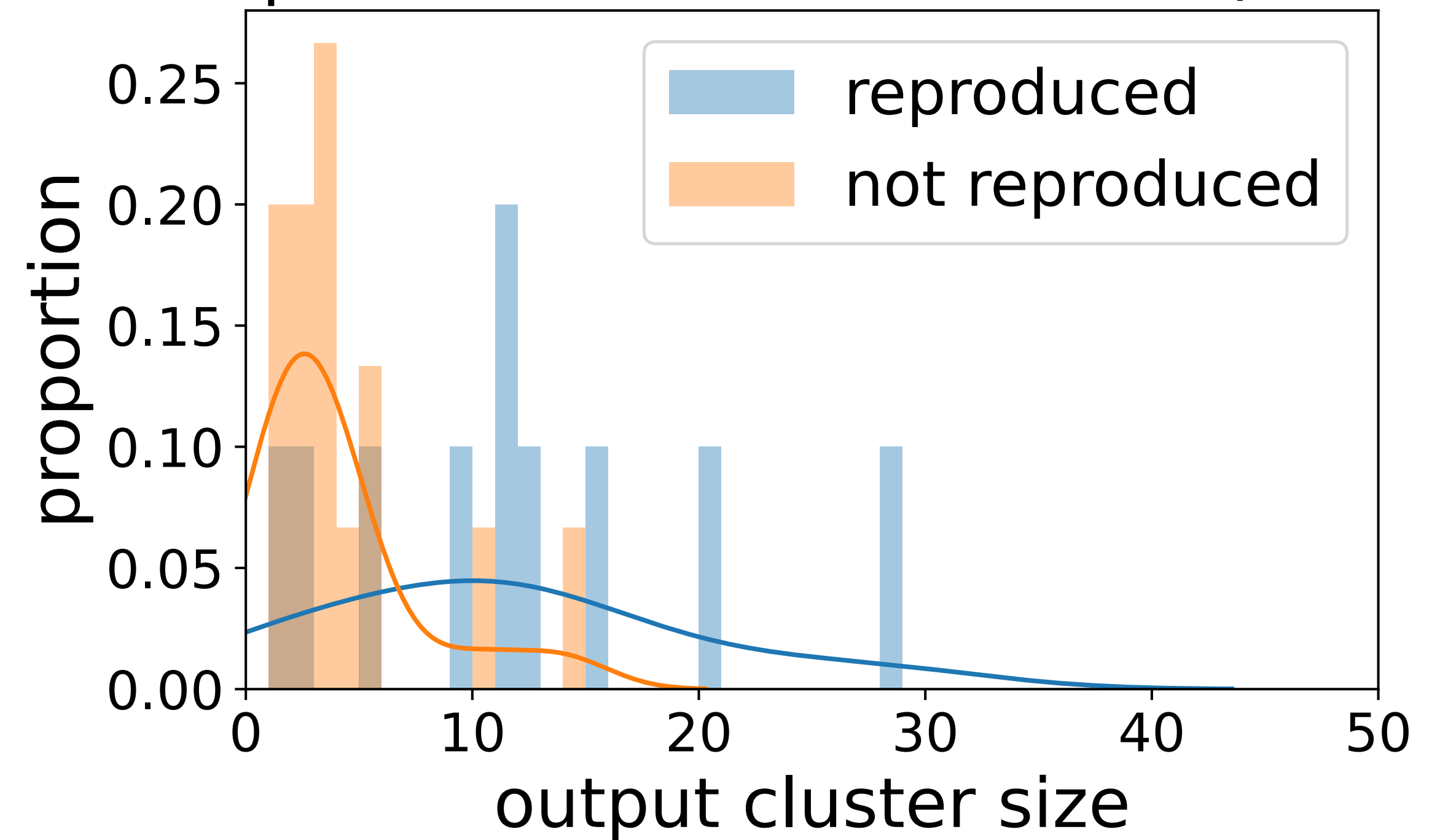


Juyeon Yoon
(PhD Candidate)

output cluster size distribution (Defects4J)



output cluster size distribution (GHRB)

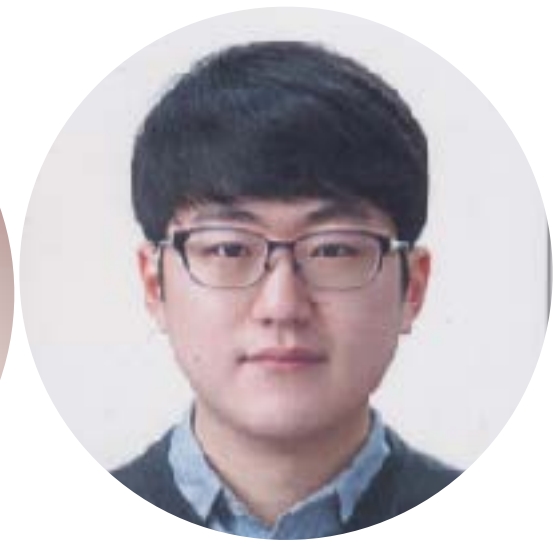


LLM-based Fault Localization

Kang, An & Yoo 2023 (<https://arxiv.org/abs/2308.05487>)

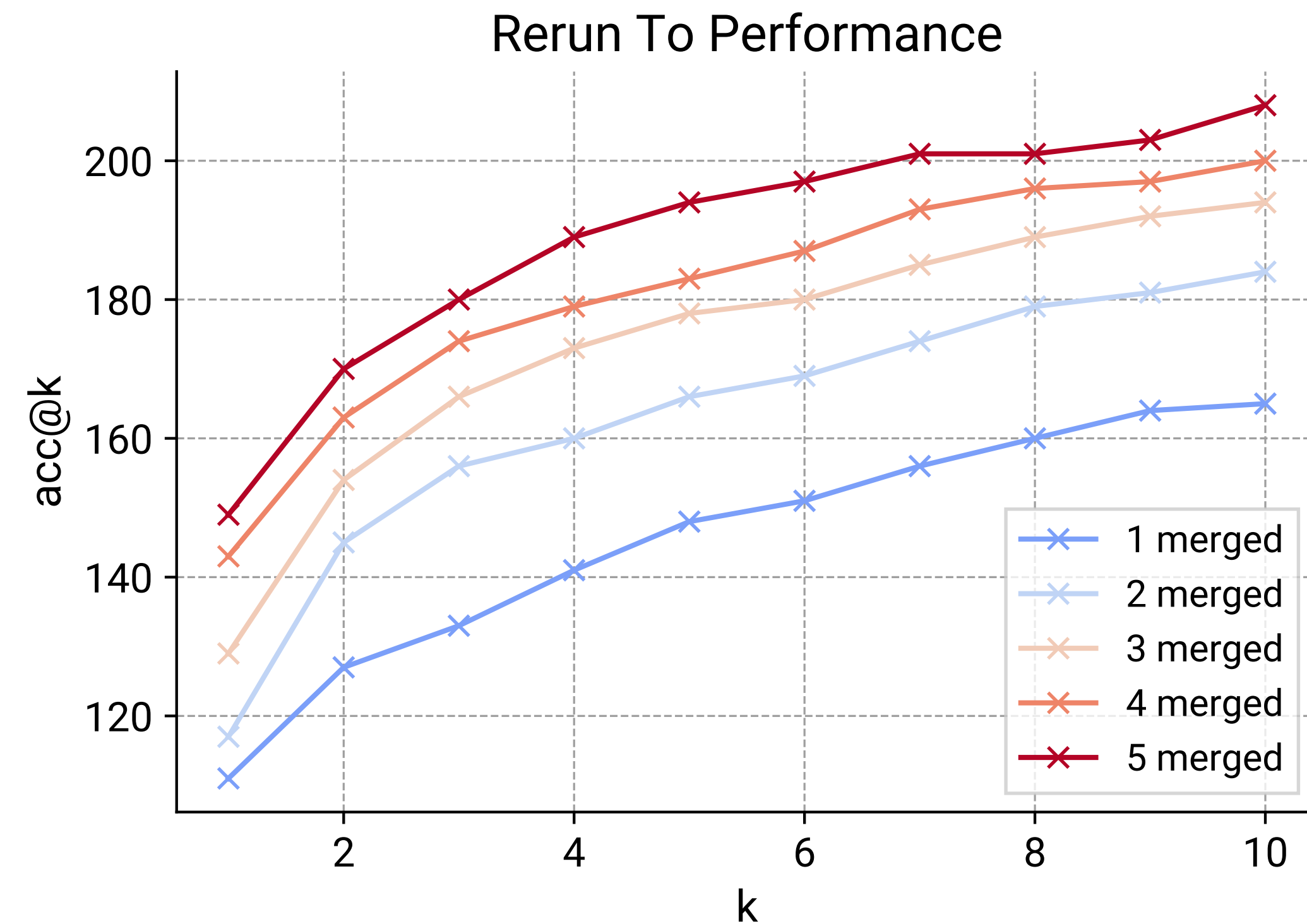


Gabin An
(PhD Candidate)



Sungmin Kang
(PhD Candidate)

Family	Technique	acc@1	acc@3	acc@5
	Predicate Switching	42	99	121
	Stack Trace	57	108	130
	Slicing (frequency)	51	96	119
MBFL	MUSE	73	139	161
	Metallaxis	106	162	191
SBFL	Ochiai	122	192	218
	DStar	125	195	216
	SBFL-F	34	66	78
LLM-Based	LLM+Test	81	94	97
	AUTOFL	149	180	194



Going Forward

Modelling self-consistency?

- The pricing model is linear to the number of tokens - self consistency is directly in conflict with the monetary computational cost.
- Would be really nice we we can model the intervals of success rate using the **problem difficulty level** and **sample size** as the input.

GPT-4

With broad general knowledge and domain expertise, GPT-4 can follow complex instructions in natural language and solve difficult problems with accuracy.

[Learn about GPT-4](#)

Model	Input	Output
gpt-4	\$0.03 / 1K tokens	\$0.06 / 1K tokens
gpt-4-32k	\$0.06 / 1K tokens	\$0.12 / 1K tokens

GPT-3.5 Turbo

GPT-3.5 Turbo models are capable and cost-effective.

`gpt-3.5-turbo-1106` is the flagship model of this family, supports a 16K context window and is optimized for dialog.

`gpt-3.5-turbo-instruct` is an Instruct model and only supports a 4K context window.

[Learn about GPT-3.5 Turbo](#) ↗

Model	Input	Output
gpt-3.5-turbo-1106	\$0.0010 / 1K tokens	\$0.0020 / 1K tokens
gpt-3.5-turbo-instruct	\$0.0015 / 1K tokens	\$0.0020 / 1K tokens

Context Length

- Some SE tasks require vast amount of background information as part of the context: sometimes this exceeds the allowed context length. What can we do?
 - In general, maximum context is getting longer (4K in gpt-3.5-turbo to 128K in gpt-4-turbo)
 - ReAct prompting can reduce the overall context by fetching what is only necessary.
 - Summarisation + vector DB: using embedding vectors as keys, you can find the context(knowledge) that is the most relevant to the current prompt and add it

LLM-based Fault Localization

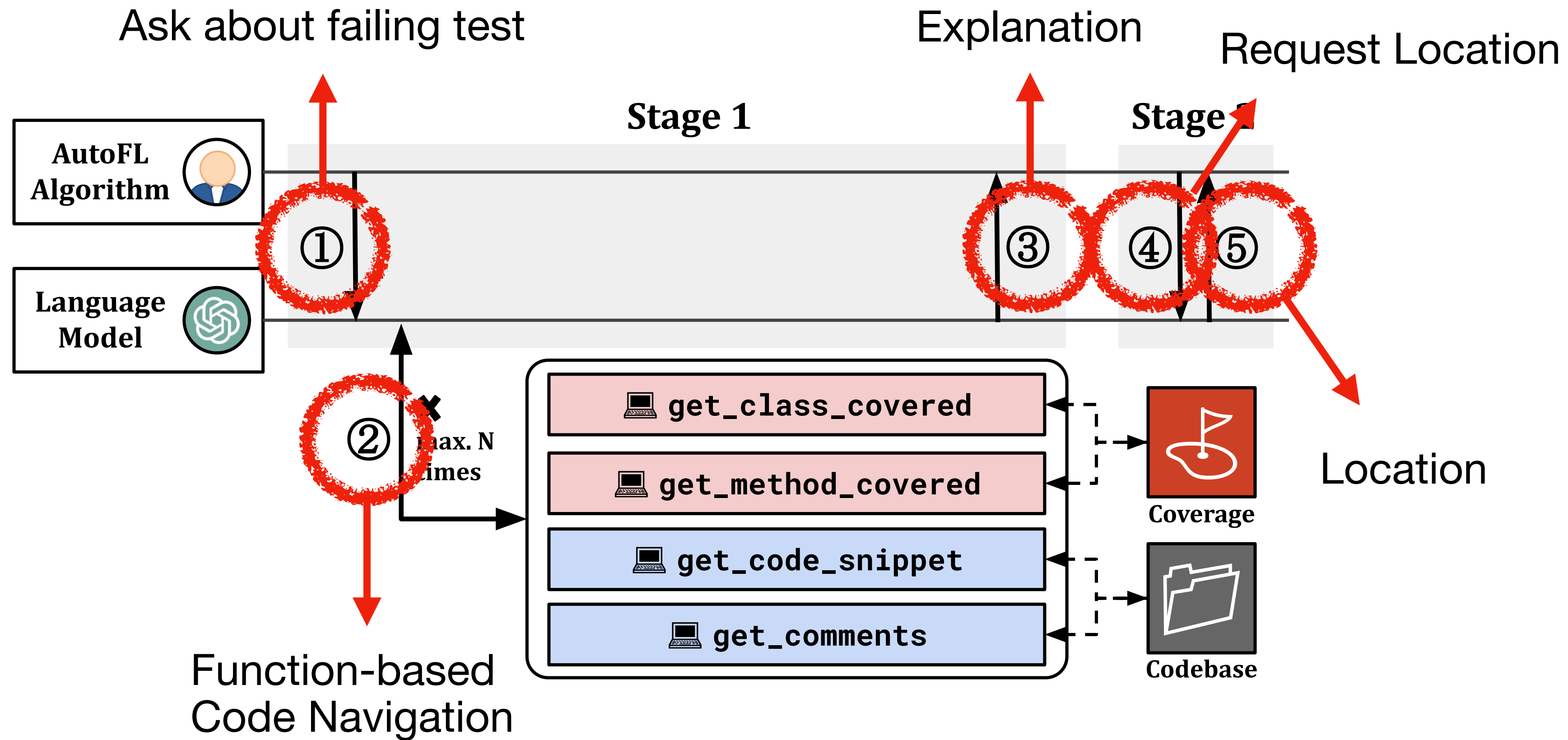
Kang, An & Yoo 2023 (<https://arxiv.org/abs/2308.05487>)



Gabin An
(PhD Candidate)

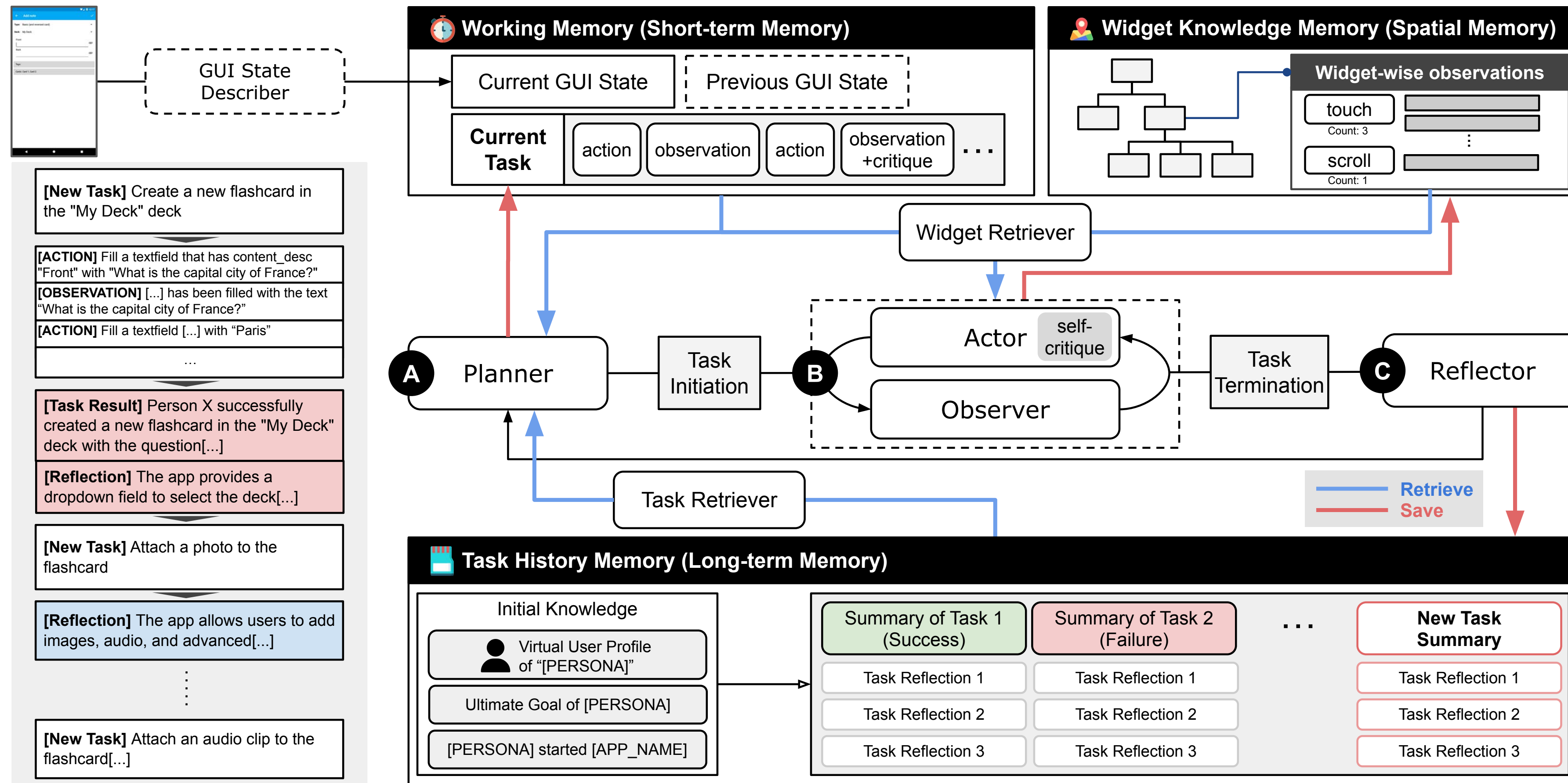


Sungmin Kang
(PhD Candidate)



Long/Short Term Memory (not LSTM...)

Yoon et al., ICST 2024 (<https://arxiv.org/abs/2311.08649>)



Juyeon Yoon
(PhD Candidate)



Prof. Robert Feldt
(Chalmers)

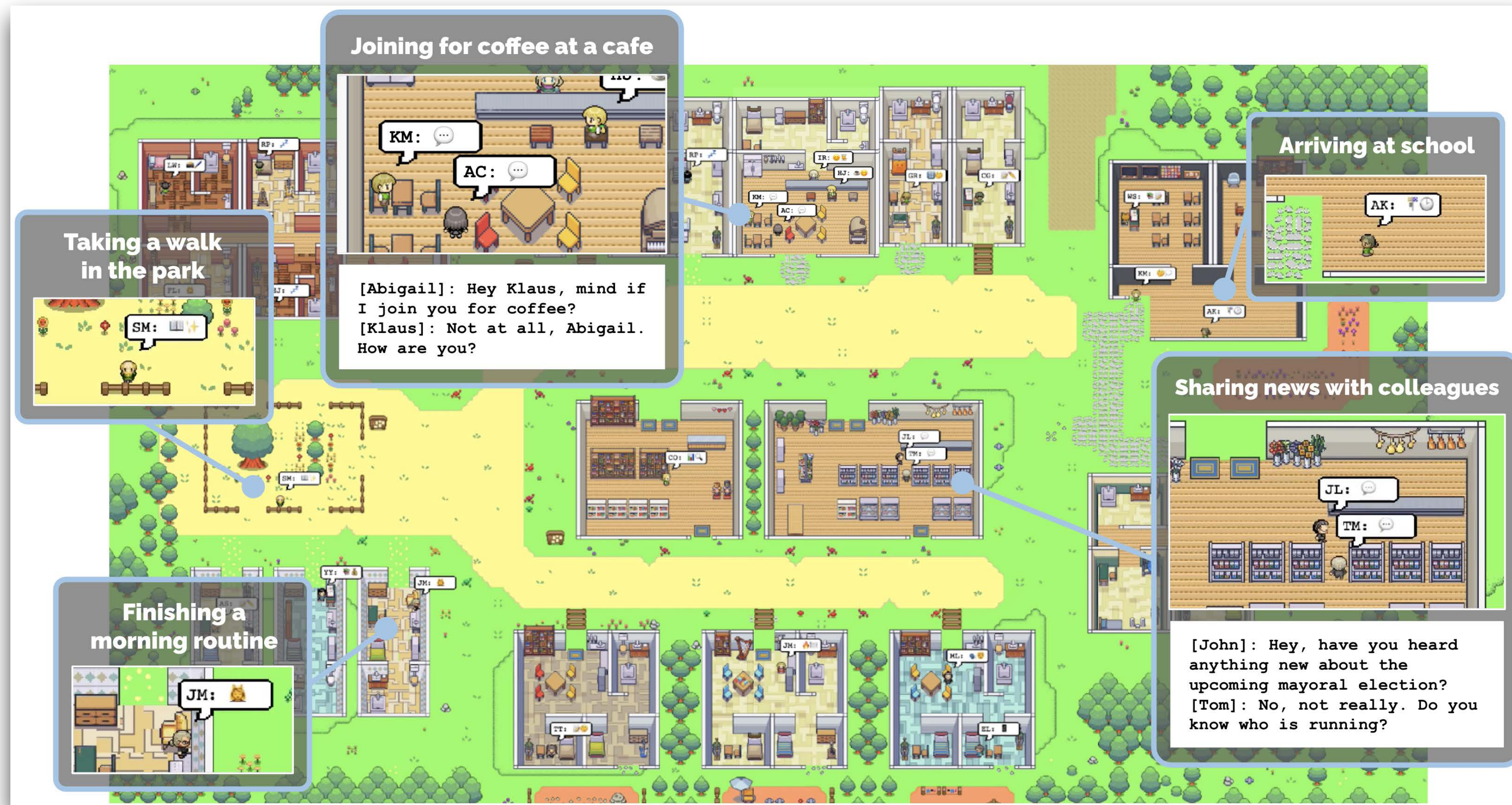
Fig. 1. Overview of DROIDAGENT with a task example.

Agency

- Currently, almost all LLM based applications consist of a single LLM instance answering to prompts.
- Classical AI literature on cognitive architecture talks of multiple modules, each partially intelligent, collaborating to form a “bigger-than-sum-of-parts” agents.
- Can we achieve higher agency by following this blueprint...?

Generative Agents: Interactive Simulacra of Human Behavior

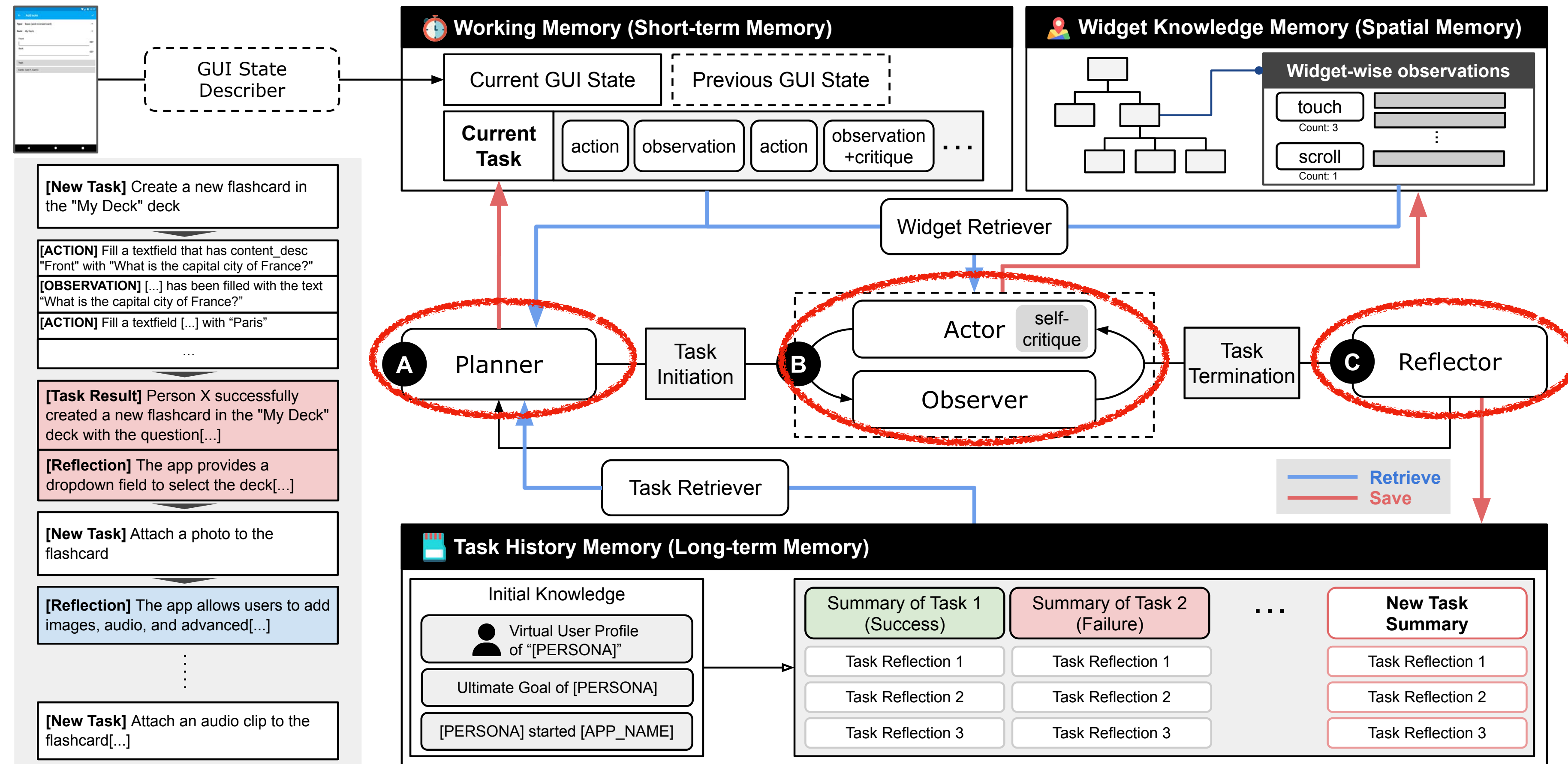
Park et al., *UIST 2023* (<https://doi.org/10.48550/arXiv.2304.03442>)



They lived happily ever after in a virtual village...

Agency

Back to Yoon et al., ICST 2024 (<https://arxiv.org/abs/2311.08649>)



Juyeon Yoon
(PhD Candidate)



Prof. Robert Feldt
(Chalmers)

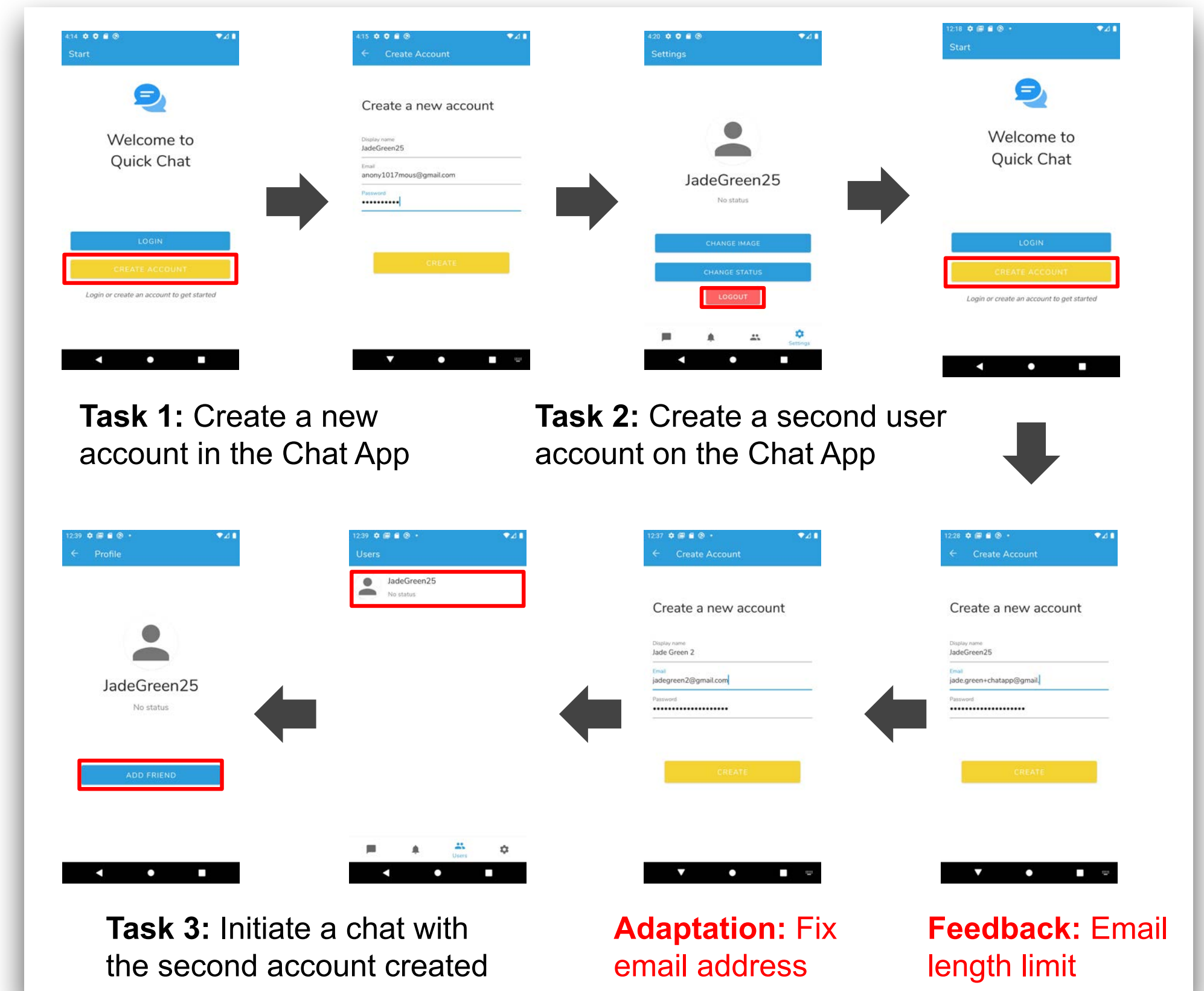
Fig. 1. Overview of DROIDAGENT with a task example.

Agency

Back to Yoon et al., ICST 2024 (<https://arxiv.org/abs/2311.08649>)

Reasoning about Jade Green's new task: To provide a diverse and realistic task that makes use of the core functionality of the app, Jade Green should try to add an audio clip to a flashcard, which is an important feature of AnkiDroid to enhance learning efficiency. This task is not too difficult as it is similar to the previous task of adding an image to a flashcard.

Jade Green's next task: Add an audio clip to a flashcard.



Juyeon Yoon
(PhD Candidate)



Prof. Robert Feldt
(Chalmers)

Disruptive Times!

- Big changes in known boundaries.
- Big changes in technical barriers.
- Ideally, we have to move beyond “I asked GPT-4 to do this and, surprise, it can do it well” now.
- Remember the Chinese Room Experiment!



Python: for _____ ...

a) i in range

b) ??

Java: for _____ ...

a) (int i = 0;

b) ??

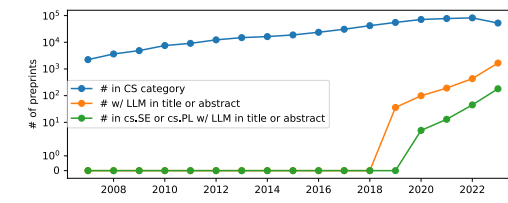


Fig. 2. Trends in number of arXiv preprints. The blue line denotes the number of preprints categorised under "CS". The orange line denotes the number of preprints in AI (cs.AD), Machine Learning (cs.LG), Neural and Evolutionary Computing (cs.NE), Software Engineering (cs.SE), and Programming Language (cs.PL) whose title or abstract contains either "Large Language Model", "LLM", or "GPT". The green line denotes the number of preprints in SE and PL categories whose title or abstract contains either "Large Language Model", "LLM", or "GPT".

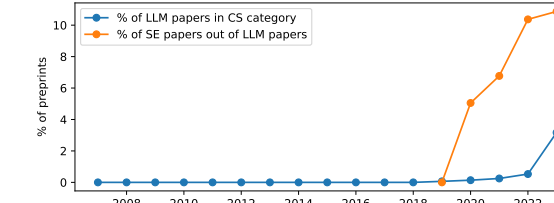
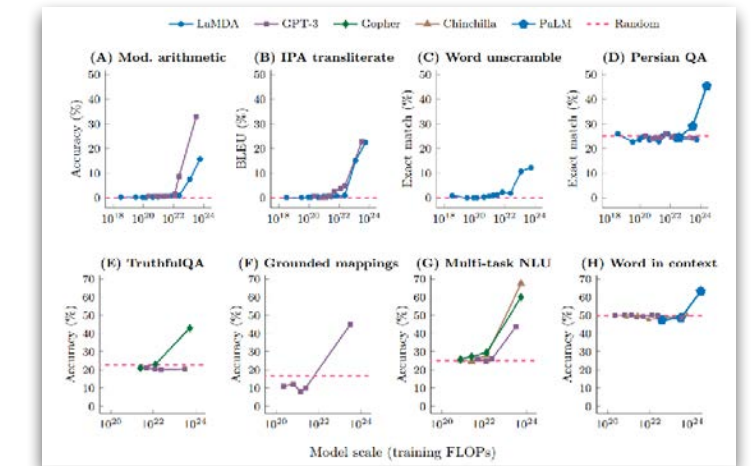


Fig. 3. Proportions of LLM papers and SE papers about LLMs. By "about LLMs", we mean that either the title or the abstract of a preprint contains "LLM", "Large Language Model", or "GPT". The blue line denotes the percentage of the number of preprints about LLMs out of the number of all preprints in the CS category. The orange line denotes the percentage of the number of preprints about LLMs in cs.SE and cs.PL categories out of all preprints about LLMs.

<https://arxiv.org/abs/2310.03533>

What is an Emergent Behavior?

- Above certain size, LLMs change their behavior in interesting ways
- The point of change in slope is referred to as "breaks"

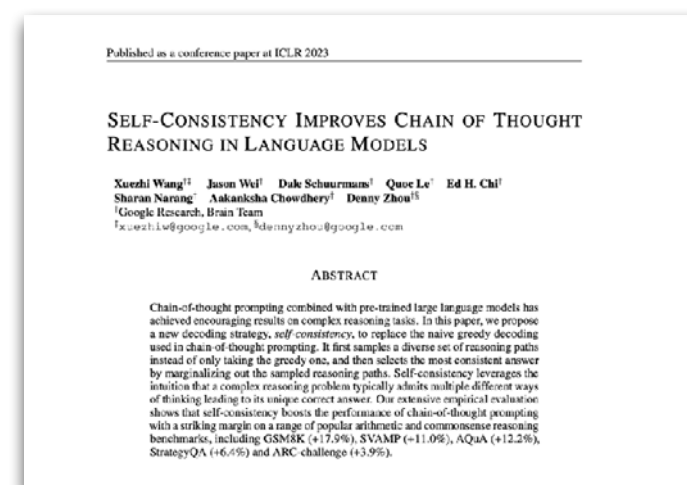


Caballero et al., <https://arxiv.org/abs/2210.14891>

Self-Consistency

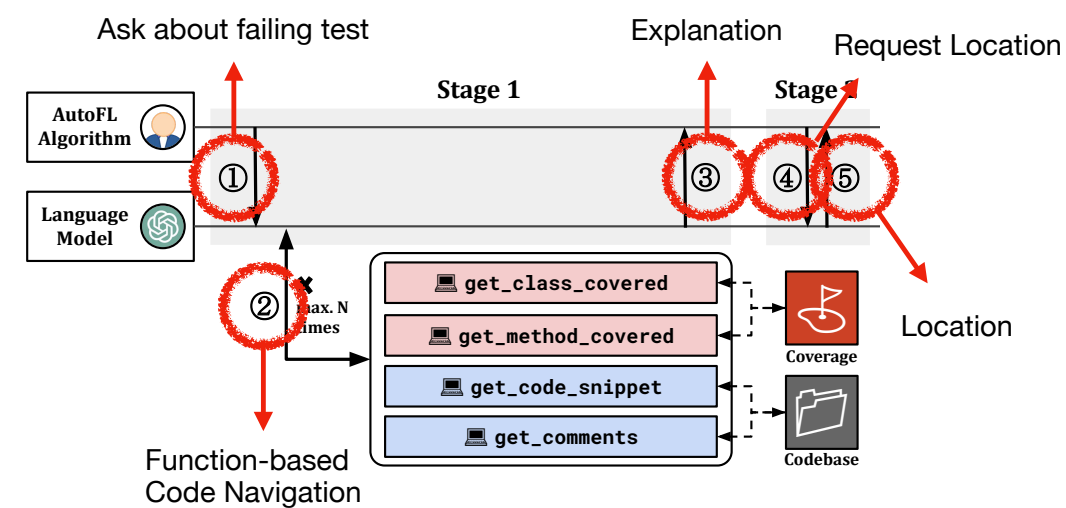
Wang et al., ICLR 2023 (<https://arxiv.org/abs/2203.11171>)

- When sampling answers from an LLM, take multiple answers with high temperature.
- If there is an answer that has the majority among the sampled answers, it is more likely to be the correct one.



LLM-based Fault Localization

Kang, An & Yoo 2023 (<https://arxiv.org/abs/2308.05487>)



Agency

Back to Yoon et al., ICST 2024 (<https://arxiv.org/abs/2311.08649>)

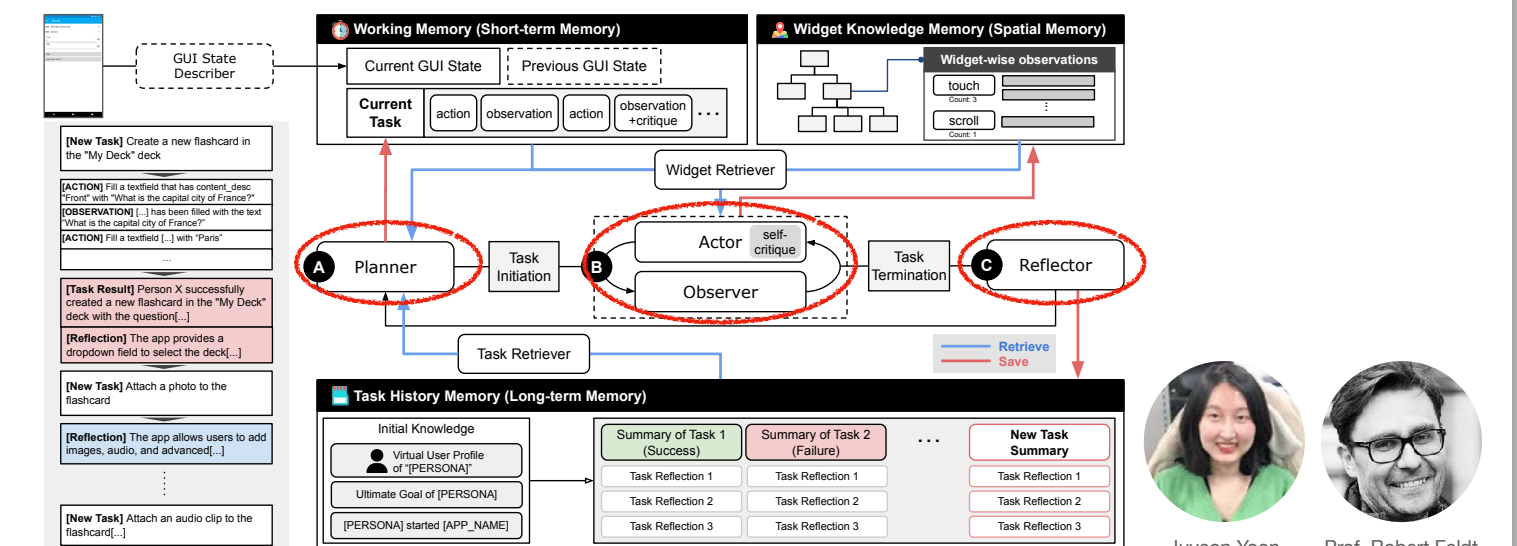


Fig. 1. Overview of DROIDAGENT with a task example.



<https://coinse.github.io>