Testing the Limits: What Breaks and How to Partially Fix LLM4ASE?

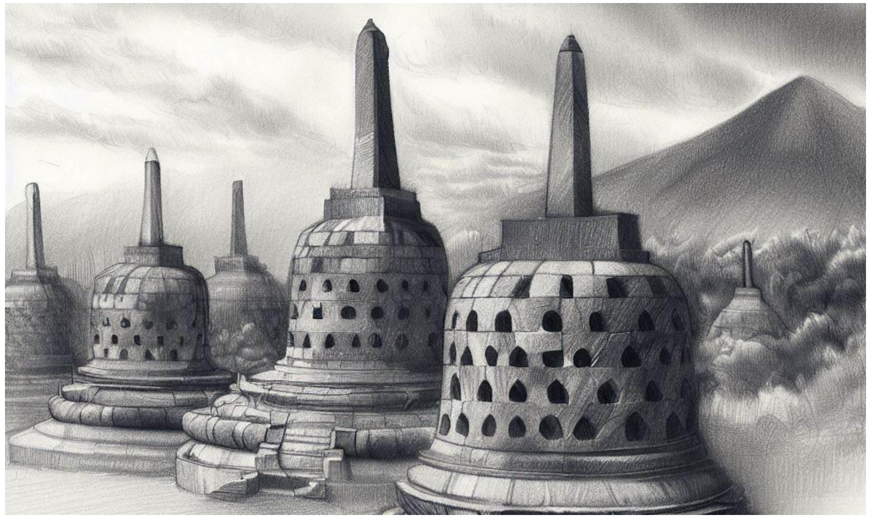
SMU
SINGAPORE MANAGEMENT UNIVERSITY

David Lo

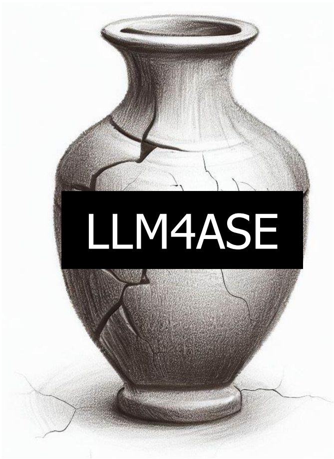
School of
Computing and
Information Systems

Testing the Limits of LLM4ASE

Why test the limits of LLM4ASE? What can we learn from history of AI4SE?

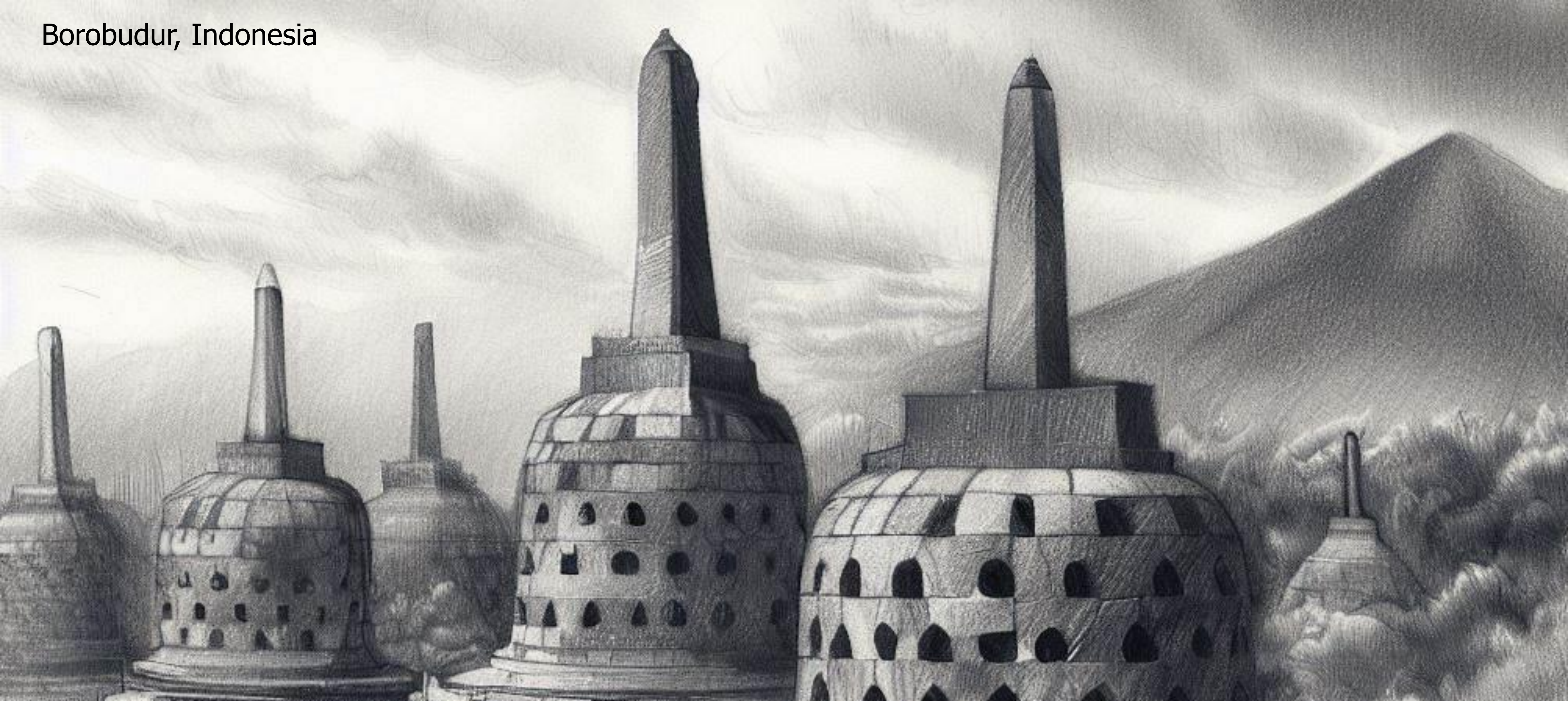


What things break when we test the limits? How to partially fix them?



What is the road ahead? What can we achieve?

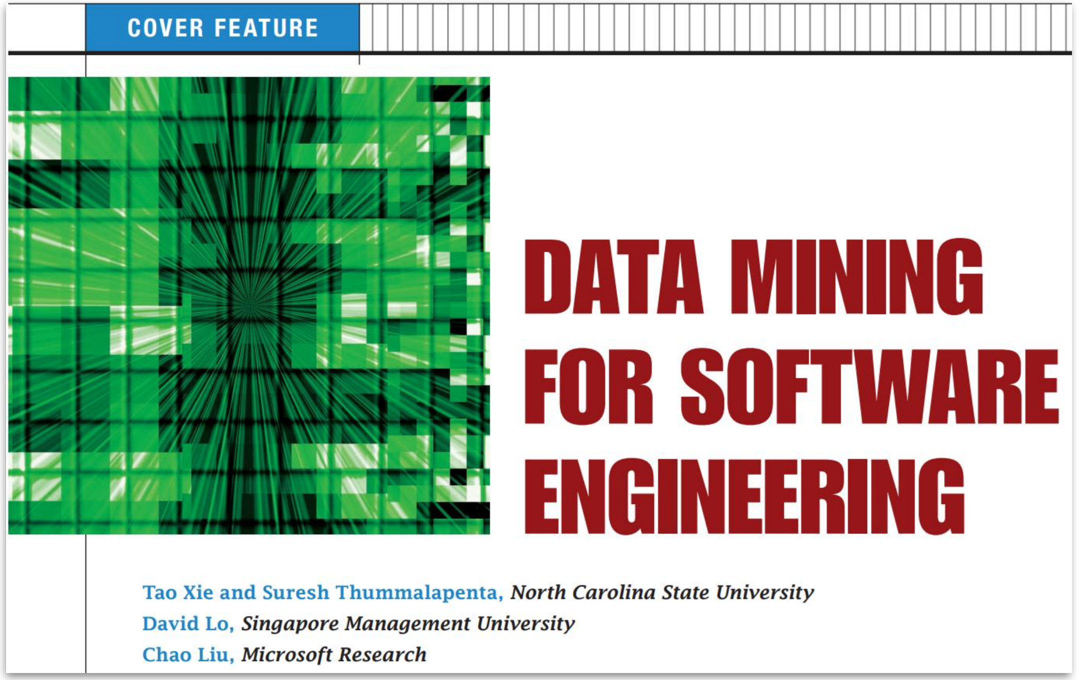
Borobudur, Indonesia



Why Test the Limits?
What Can We Learn from AI4SE History?

AI for Software Engineering (AI4SE)

MSR 2004: International Workshop on Mining Software Repositories
2004.msrconf.org


COVER FEATURE

DATA MINING FOR SOFTWARE ENGINEERING

Tao Xie and Suresh Thummalapenta, *North Carolina State University*
David Lo, *Singapore Management University*
Chao Liu, *Microsoft Research*

IEEE Computer 2009

AI for Software Engineering (AI4SE)

Andreas Zeller @AndreasZeller · Mar 31, 2022

Machine learning rules! Machine Learning with and for Software Engineering is now the #1 topic in terms of submitted papers, before all-time favorites such as software testing and analysis.

Top 10 Topics – Submitted

Topics	# Submitted Papers	# Accepted Papers	Acceptance Rate
Machine Learning with and for SE	237	74	31,22%
Software Testing	181	47	25,97%
Program Analysis	117	35	29,91%
Evolution and maintenance	105	31	29,52%
Mining Software Repositories	105	23	21,90%
Software Security	85	25	29,41%
Human Aspects of SE	68	20	29,41%
Validation and Verification	53	15	28,30%
Tools and Environments	49	12	24,49%
Reliability and Safety	46	15	32,61%

Wave 1: Deep Learning

Toward Deep Learning Software Repositories

Martin White, Christopher Vendome, Mario Linares-Vásquez, and Denys Poshyvanyk
Department of Computer Science
College of William and Mary
Williamsburg, Virginia 23187–8795
Email: {mgwhite, cvendome, mlinarev, denys}@cs.wm.edu

MSR 2015

Deep Learning for Just-In-Time Defect Prediction

Xinli Yang[*], David Lo[†], Xin Xia[*‡], Yun Zhang[*], and Jianling Sun[*]
[*]College of Computer Science and Technology, Zhejiang University, Hangzhou, China
[†]School of Information Systems, Singapore Management University, Singapore
zdyxl@zju.edu.cn, davidlo@smu.edu.sg, {xxia, yunzhang28, sunjl}@zju.edu.cn

QRS 2015

Wave 1: Deep Learning

Key Question: How Can We Learn Better Representations of Code and Other Software Artifacts?

Deep Belief Network (DBN)

Convolutional Neural Network (CNN)

Recurrent Neural Network (RNN)

Long Short-Term Memory Network (LSTM)

Transformer

...

Wave 1: Deep Learning

Despite the wave, answer is **not always positive**

FSE 2017

Easy over Hard: A Case Study on Deep Learning

Wei Fu, Tim Menzies
Com.Sci., NC State, USA
wfu@ncsu.edu,tim.menzies@gmail.com

Simple methods can do better than a deep learning approach for mining SO

ISSTA 2021

Deep Just-in-Time Defect Prediction: How Far Are We?

Zhengran Zeng
Southern University of Science and Technology
China
12032889@mail.sustech.edu.cn

Yuqun Zhang*
Southern University of Science and Technology
China
zhangyq@sustech.edu.cn

Haotian Zhang
Kwai Inc.
China
zhanghaotian@kuaishou.com

Lingming Zhang
University of Illinois Urbana-Champaign
United States
lingming@illinois.edu

Simple methods can do better than DL approaches for JIT defect-prediction

Wave 1: Deep Learning

Finding the limits (weaknesses) can result in **new innovations**

FSE 2017

Easy over Hard: A Case Study on Deep Learning

Wei Fu, Tim Menzies
Com.Sci., NC State, USA
wfu@ncsu.edu,tim.menzies@gmail.com

ESEM 2018

Prediction of Relatedness in Stack Overflow: Deep Learning vs. SVM

A Reproducibility Study

Bowen Xu*
Singapore Management University
bowenxu.2017@smu.edu.sg

Amirreza Shirani*
University of Houston
ashirani@uh.edu

David Lo

Mohammad Amin Alipour

Improve dataset, create new SOTA baseline

Highly Commended Paper Award (Best Paper Runner Up)

Wave 1: Deep Learning

Finding the limits (weaknesses) can result in **new innovations**

ISSTA 2021

Deep Just-in-Time Defect Prediction: How Far Are We?

Zhengran Zeng
Southern University of Science and Technology
China
12032889@mail.sustech.edu.cn

Yuqun Zhang*
Southern University of Science and Technology
China
zhangyq@sustech.edu.cn

Haotian Zhang
Kwai Inc.
China
zhanghaotian@kuaishou.com

Lingming Zhang
University of Illinois Urbana-Champaign
United States
lingming@illinois.edu

ICPC 2022

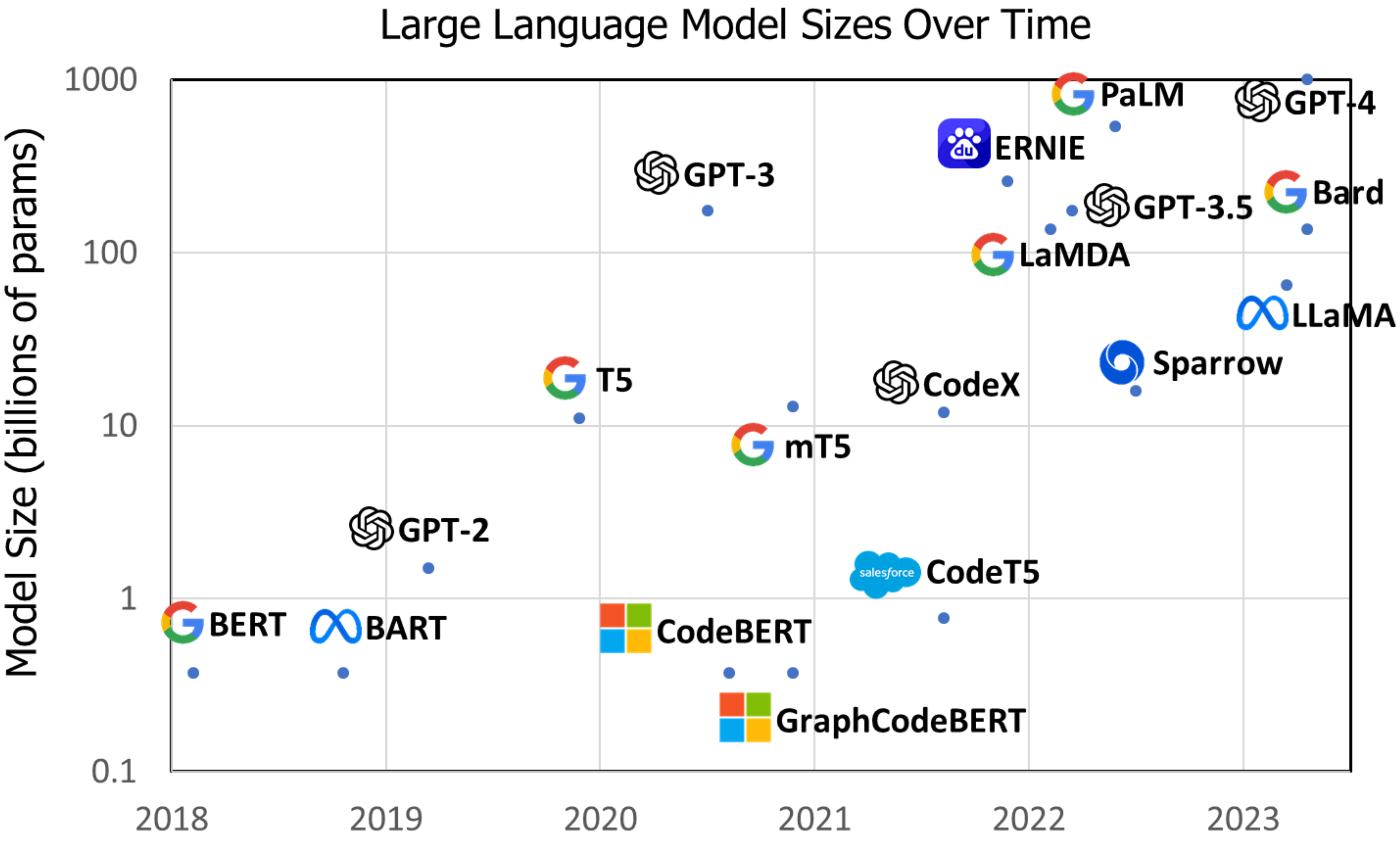
Simple or Complex? Together for a More Accurate Just-In-Time Defect Predictor

Xin Zhou, DongGyun Han, and David Lo
School of Computing and Information Systems, Singapore Management University
Singapore
xinzhou.2020@phdcs.smu.edu.sg,{dhan,davidlo}@smu.edu.sg

Combine deep learning and simple methods to achieve SOTA

Invited for EMSE extension

Wave 2: Large Language Models (LLMs)



Language Model Potentials

ICSE 2012

On the Naturalness of Software

Abram Hindle, Earl Barr, Mark Gabel, Zhendong Su, Prem Devanbu
devanbu@cs.ucdavis.edu

> *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.*

Won Most Influential Paper Award @ ICSE 2022

Our Initial Experience with LLMs

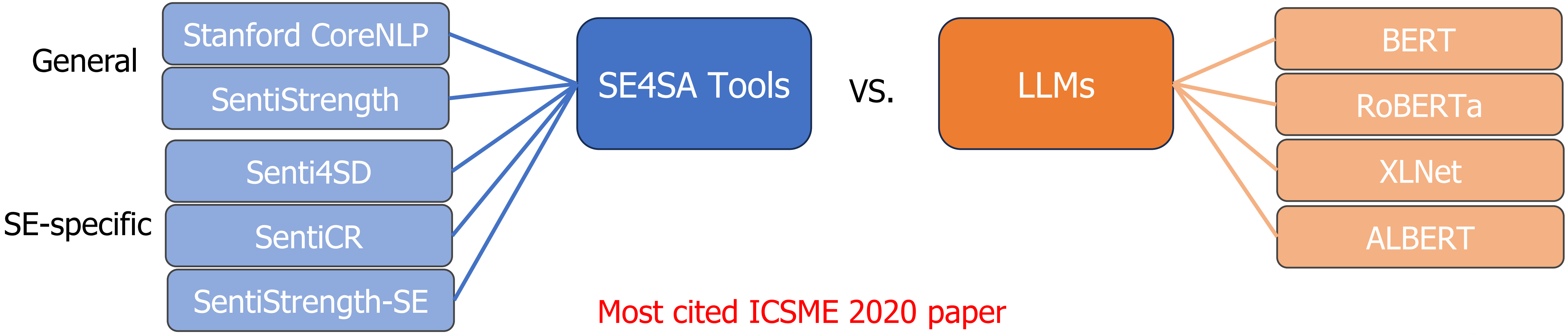
ICSME 2020

Sentiment Analysis for Software Engineering: How Far Can Pre-trained Transformer Models Go?

Ting Zhang, Bowen Xu*, Ferdian Thung, Stefanus Agus Haryono, David Lo, Lingxiao Jiang
School of Information Systems, Singapore Management University
Email: {tingzhang.2019, bowenxu.2017}@phdcs.smu.edu.sg, {ferdianthung, stefanusah, davidlo, lxjiang}@smu.edu.sg

Is LLM a silver bullet or hype for software engineering?

- Consider a common task (NLP and SE): sentiment analysis
- Many specialized techniques have been proposed for SE



Most cited ICSME 2020 paper

Our Initial Experience with LLMs

ICSME 2020

Sentiment Analysis for Software Engineering: How Far Can Pre-trained Transformer Models Go?

Ting Zhang, Bowen Xu*, Ferdian Thung, Stefanus Agus Haryono, David Lo, Lingxiao Jiang
School of Information Systems, Singapore Management University
Email: {tingzhang.2019, bowenxu.2017}@phdcs.smu.edu.sg, {ferdianthung, stefanusah, davidlo, lxjiang}@smu.edu.sg

Is LLM a silver bullet or hype for software engineering?

SE4SA Tools vs. LLMs

Our Initial Experience with LLMs

ICSME 2020

Sentiment Analysis for Software Engineering: How Far Can Pre-trained Transformer Models Go?

Ting Zhang, Bowen Xu*, Ferdian Thung, Stefanus Agus Haryono, David Lo, Lingxiao Jiang
School of Information Systems, Singapore Management University
Email: {tingzhang.2019, bowenxu.2017}@phdcs.smu.edu.sg, {ferdianthung, stefanusah, davidlo, lxjiang}@smu.edu.sg

Is LLM a silver bullet or hype for software engineering?

Metric	Group	API	SO	App	GitHub	Jira	CR
Macro-avg F1	Best **PRIOR**	0.66	0.59	0.53	0.82	0.91	0.72
	Best **PTM**	0.82	0.80	0.61	0.92	0.98	0.84
	Improvement	24.2%	35.6%	15.1%	12.2%	7.7%	16.7%
Micro-avg F1	Best **PRIOR**	0.82	0.83	0.77	0.83	0.92	0.78
	Best **PTM**	0.89	0.90	0.88	0.92	0.98	0.88
	Improvement	8.5%	8.4%	14.3%	10.8%	6.5%	12.8%

Our Initial Experience with LLMs

ICSME 2021

Assessing Generalizability of CodeBERT

Xin Zhou, DongGyun Han, and David Lo
School of Computing and Information Systems, Singapore Management University
xinzhou.2020@phdcs.smu.edu.sg, {dhan, davidlo}@smu.edu.sg

Is LLM a silver bullet or hype for software engineering?

- CodeBERT (EMNLP'20) was not evaluated beyond its pre-trained dataset
- Only shown effective on 2 tasks and not compared with SE SOTA
- Can it generalize to **additional data**, **task,** and **baselines**?
- *Answer*: Yes, Yes, Yes -- by 3.7% (new task) to 31% (old task, new data)
- *Tradeoff*: Much more computation resources
  - CodeBERT is 9-24x slower than NCS and UNIF (10k documents)
  - Gap is bigger for larger corpus

Most cited ICSME 2021 paper

LLMs Seem to Win for Many ASE Scenarios

ASE 2021

Finding A Needle in a Haystack: Automated Mining of Silent Vulnerability Fixes

Jiayuan Zhou*, Michael Pacheco*, Zhiyuan Wan†, Xin Xia‡∥, David Lo§, Yuan Wang* and Ahmed E. Hassan¶

Finding silent vulnerability fixes

ICPC 2022

Benchmarking Library Recognition in Tweets

Ting Zhang, Divya Prabha Chandrasekaran, Ferdian Thung, David Lo
School of Computing and Information Systems, Singapore Management University
{tingzhang.2019,divyaprabha.2021,ferdianthung,davidlo}@smu.edu.sg

Mining social media for library review and rant

LLMs Seem to Win for Many ASE Scenarios

FSE 2022

AutoPruner: Transformer-Based Call Graph Pruning

Thanh Le-Cong
Hong Jin Kang
Truong Giang Nguyen
Stefanus Agus Haryono
David Lo

Xuan-Bach D. Le
University of Melbourne
Melbourne, Victoria, Australia

Quyet Thang Huynh
Hanoi University of Science and Technology
Hanoi, Vietnam

Neurosymbolic analysis to deal with imprecision of static analysis

TSE 2023

Invalidator: Automated Patch Correctness Assessment via Semantic and Syntactic Reasoning

Thanh Le-Cong, Duc-Minh Luong, Xuan Bach D. Le, David Lo,
Nhat-Hoa Tran, Bui Quang-Huy and Quyet-Thang Huynh

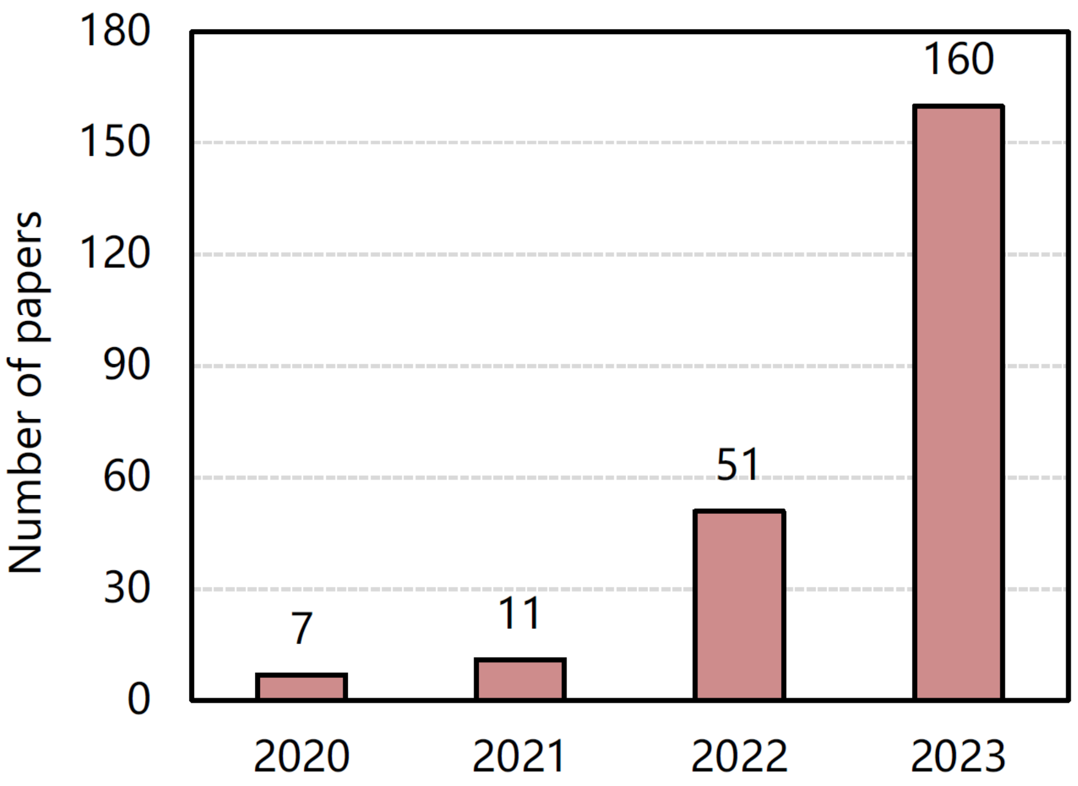
Neurosymbolic analysis to determine patch correctness

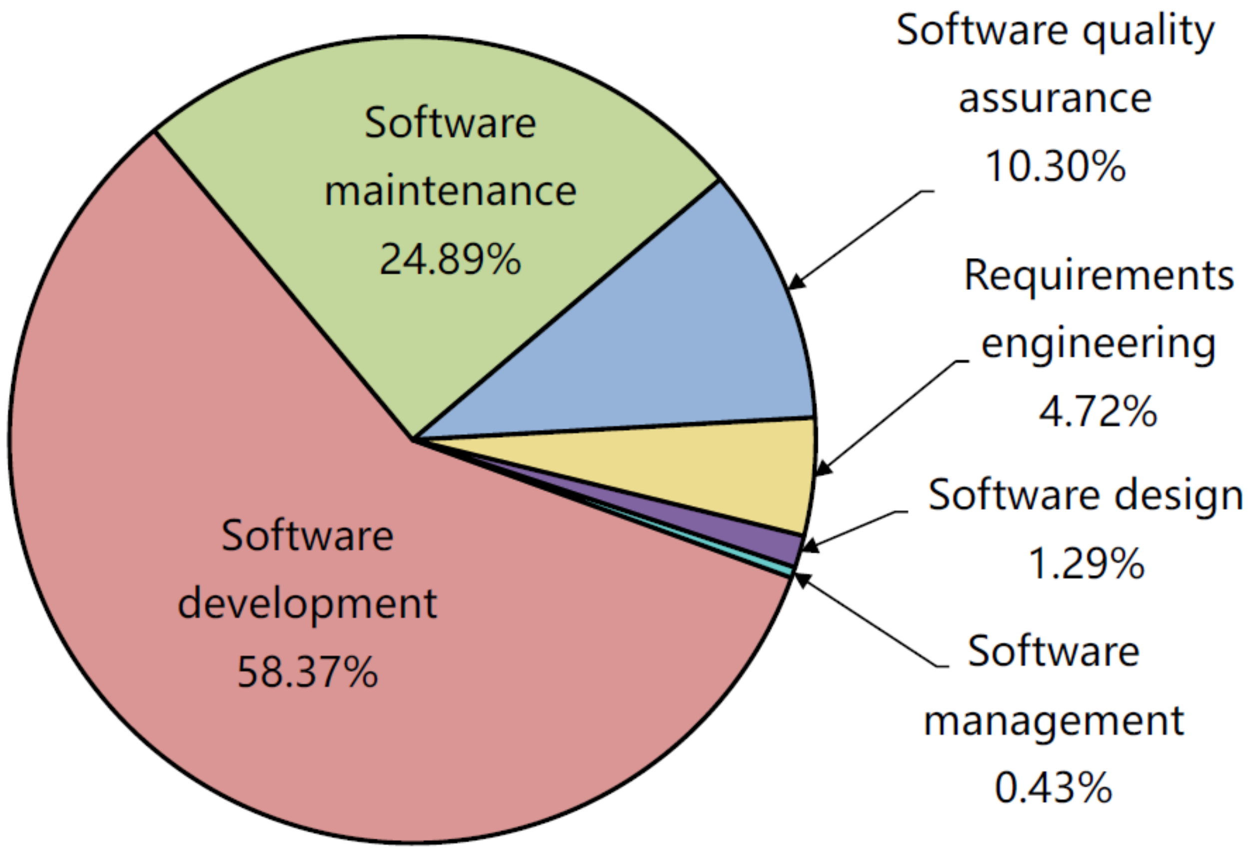
LLMs Seem to Win for Many ASE Scenarios

Large Language Models for Software Engineering: A Systematic Literature Review

XINYI HOU*, Huazhong University of Science and Technology, China
YANJIE ZHAO*, Monash University, Australia
YUE LIU, Monash University, Australia
ZHOU YANG, Singapore Management University, Singapore
KAILONG WANG, Huazhong University of Science and Technology, China
LI LI, Beihang University, China
XIAPU LUO, The Hong Kong Polytechnic University, China
DAVID LO, Singapore Management University, Singapore
JOHN GRUNDY, Monash University, Australia
HAOYU WANG†, Huazhong University of Science and Technology, China

LLMs Seem to Win for Many ASE Scenarios

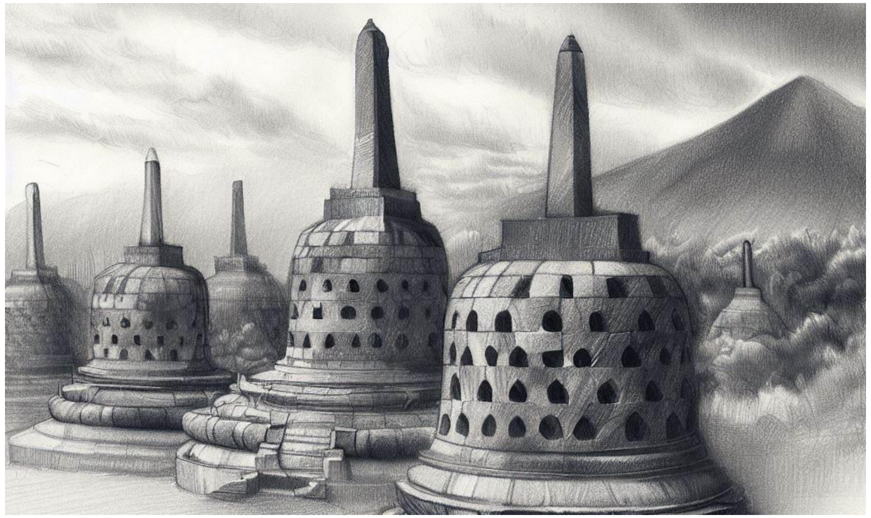




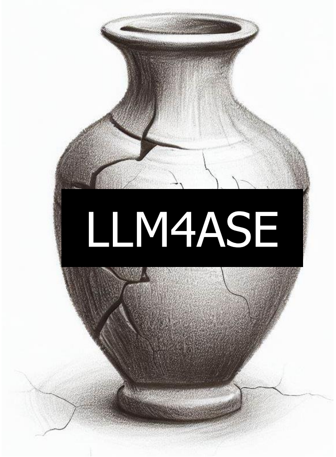
Have we tested the limits of LLM4ASE?

Testing the Limits of LLM4ASE

Why test the limits of LLM4ASE? What can we learn from history of AI4SE?



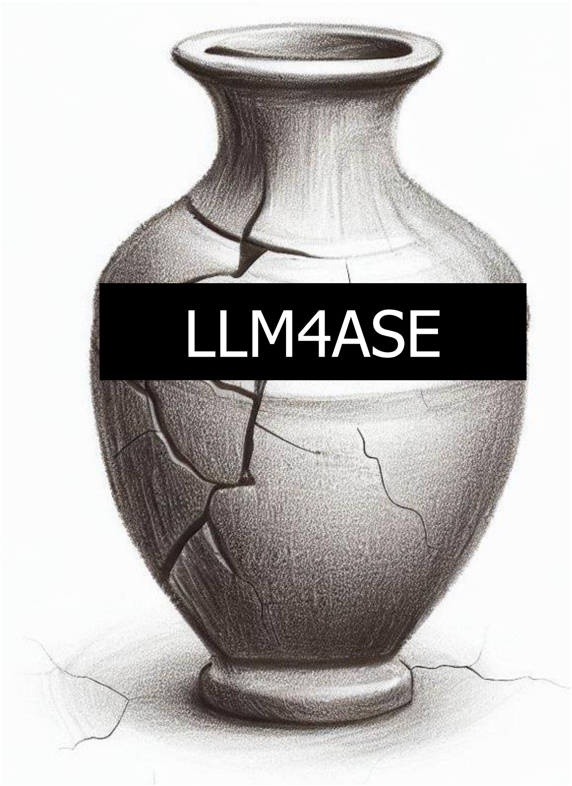
What things break when we test the limits? How to partially fix them?



What is the road ahead? What can we achieve?

What Things Break? How to Partially Fix Them?

Data



Long-Tailed Data

Data Evolution

Backdoor

Form over Content

Size & Latency

Robustness

Hallucination

Model

Data

Long-Tailed Data

LLM4ASE Performs Badly on Tail Data

ASE 2023

The Devil is in the Tails: How Long-Tailed Code Distributions Impact Large Language Models

Xin Zhou[†], Kisub Kim[*†], Bowen Xu[†‡], Jiakun Liu[†], DongGyun Han[§], David Lo[†]
[†]Singapore Management University, Singapore
{xinzhou.2020, bowenxu.2017}@phdcs.smu.edu.sg, {kisubkim, jkliu, davidlo}@smu.edu.sg
[‡]North Carolina State University, USA
bxu22@ncsu.edu
[§]Royal Holloway, University of London, UK
donggyun.han@rhul.ac.uk

Experiment Design

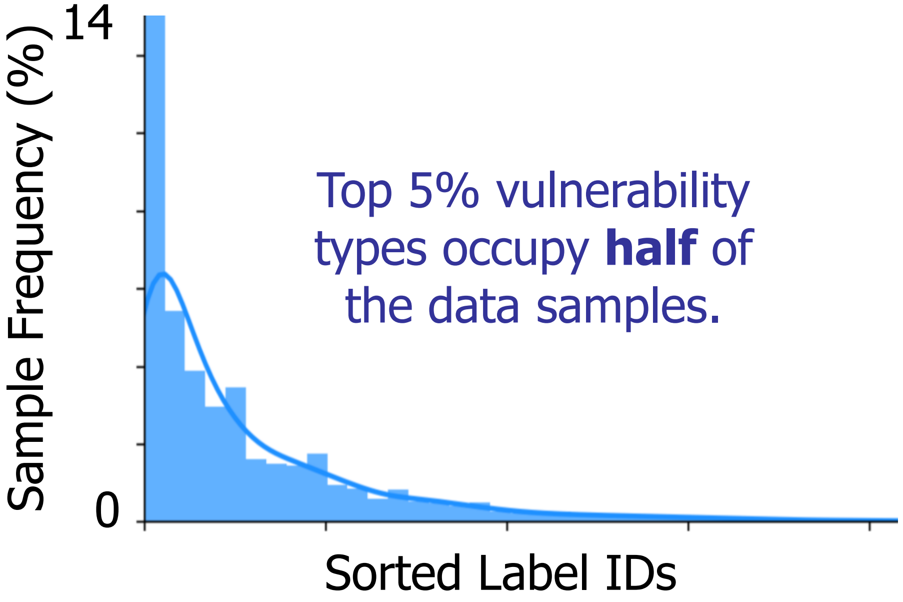
How does LLM4ASE perform on long-tailed data?

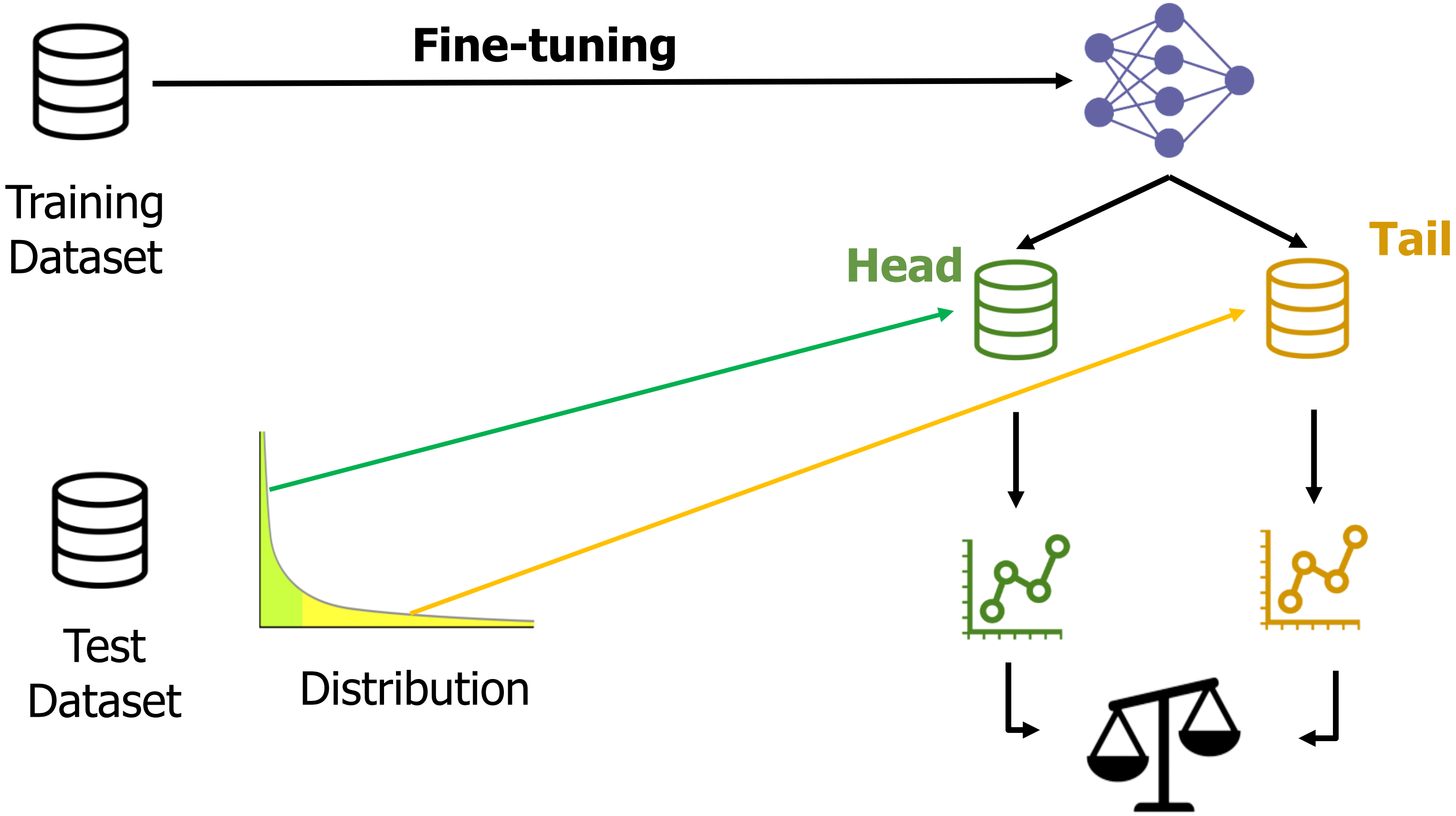
1. Long Tailed Data
2. Fine-tuning
3. Evaluation on Head and Tail

API Sequence Recommendation

Code Revision Recommendation

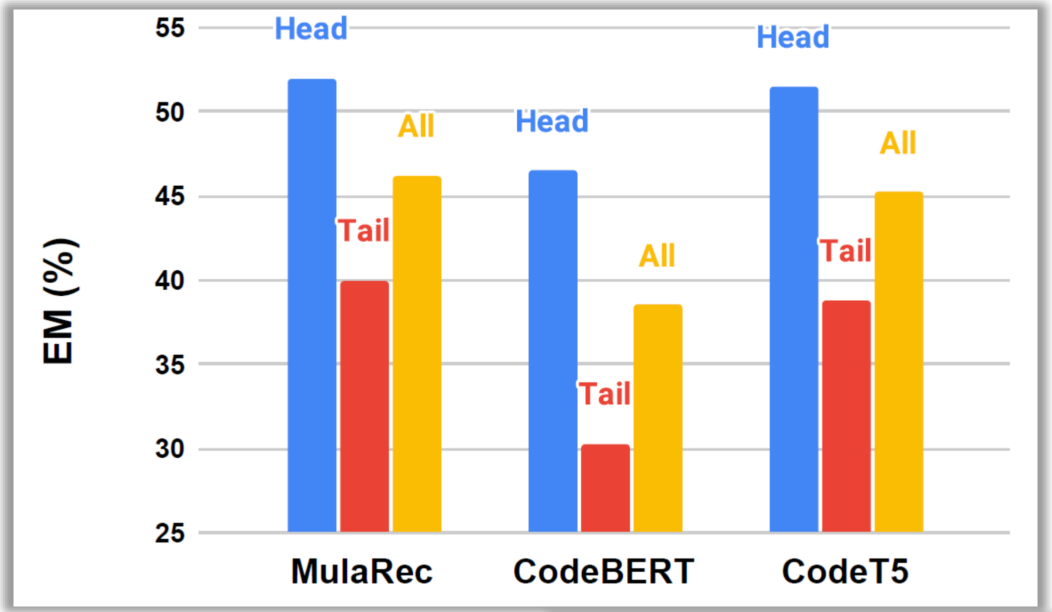
Vulnerability Type Prediction



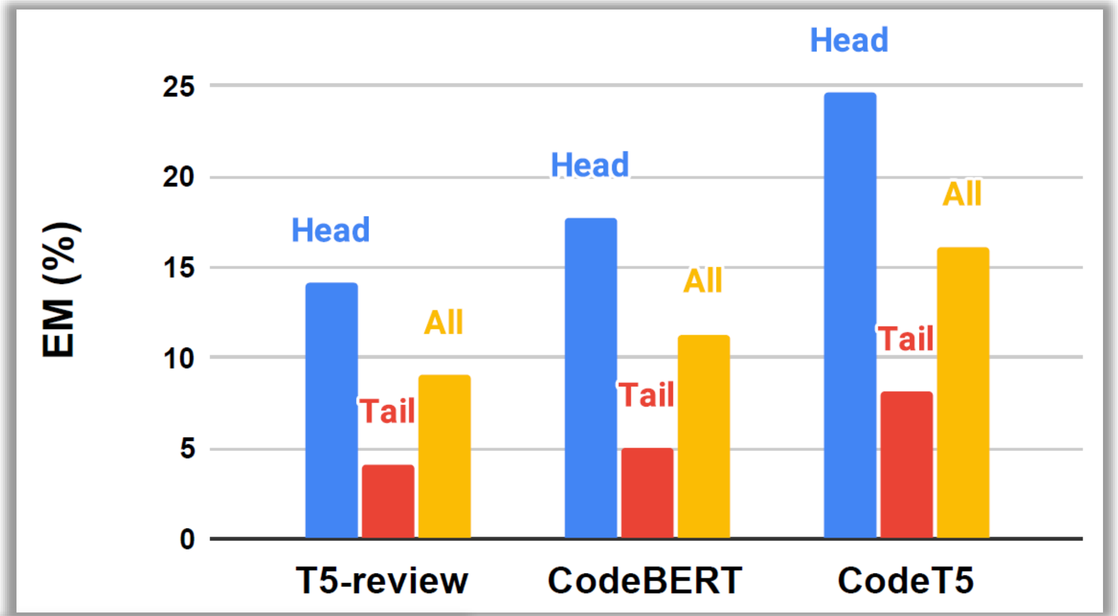


Results

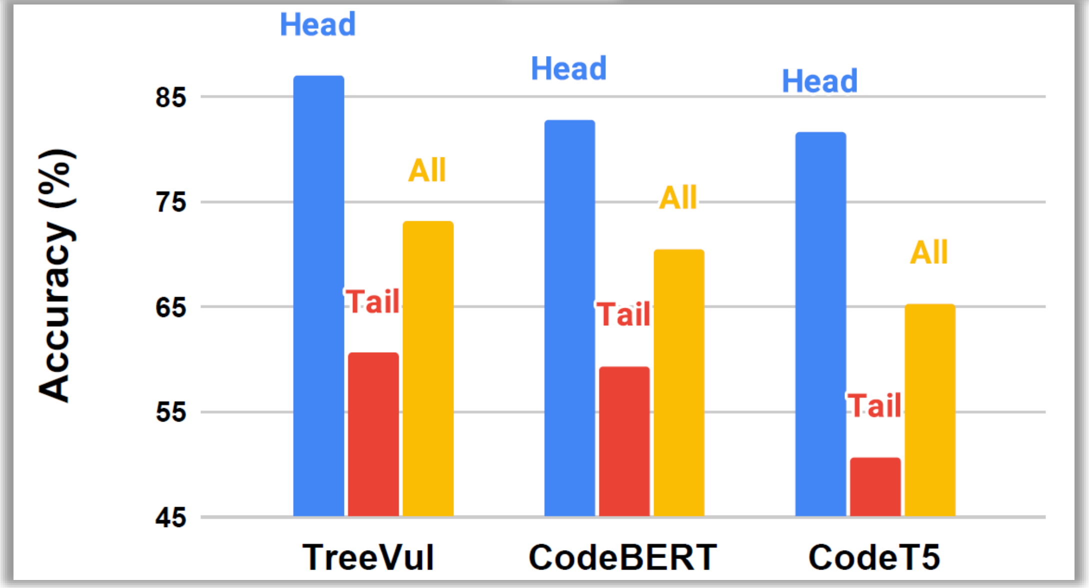
(a) API Sequence Rec.



(b) Code Revision Rec.

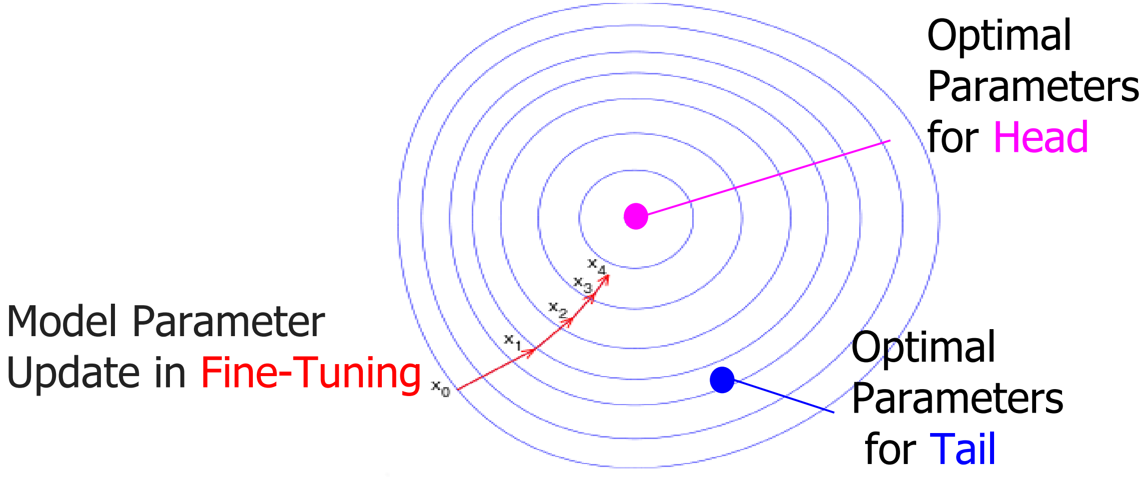


(c) Vulnerability Type Rec.

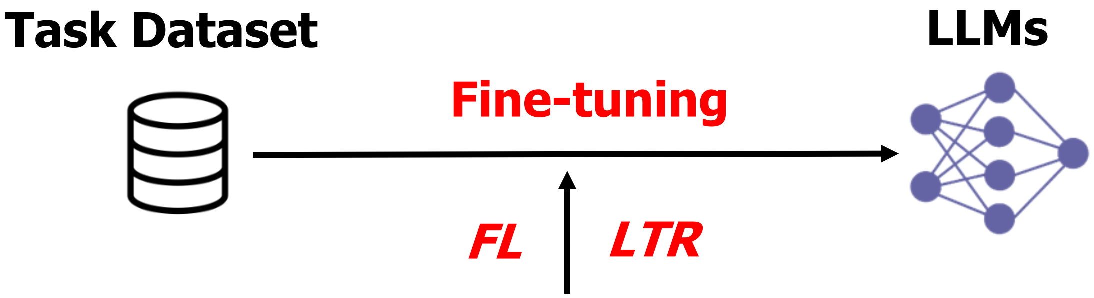


30-200% Difference in Head vs. Tail

Mitigation



Fine-tuning is biased by frequent head data



Assigning more weight to tail samples during parameter updates

Effectiveness

Vulner. Type	Head	Tail	All
TreeVul	**87.0**	60.6	73.1
+ FL[30]	82.8 (4.2% ↓)	59.4 (1.2% ↓)	70.5 (2.6% ↓)
+ LTR[31]	**87.0** (0.0% —)	61.2 (0.6% ↑)	**73.4** (0.3% ↑)
CodeBERT	82.8	59.4	70.5
+ FL[30]	81.5 (1.3% ↓)	**61.7** (2.3% ↑)	71.1 (0.6% ↑)
+ LTR[31]	82.8 (0.0% —)	60.3 (0.9% ↑)	70.9 (0.4% ↑)
CodeT5	81.6	50.7	65.3
+ FL[30]	80.3 (1.3% ↓)	53.4 (2.7% ↑)	65.9 (0.6% ↑)
+ LTR[31]	80.3 (1.3% ↓)	54.5 (3.8% ↑)	66.7 (1.4% ↑)

Vulnerability Type Prediction (Accuracy)

Mitigation techniques have the potential to improve LLMs' handling of tails, although the effectiveness is is limited.

Data Evolution

LLMs are Affected by Data Evolution

FSE 2023

CL4Code: Leveraging Continual Learning for Out-of-Distribution API Prediction

Martin Weyssow
martin.weyssow@umontreal.ca
Université de Montréal

Xin Zhou
xinzhou.2020@phdcs.smu.edu.sg
Singapore Management University

Kisub Kim
kisubkim@smu.edu.sg
Singapore Management University

David Lo
davidlo@smu.edu.sg
Singapore Management University

Houari Sahraoui
sahraouh@iro.umontreal.ca
Université de Montréal

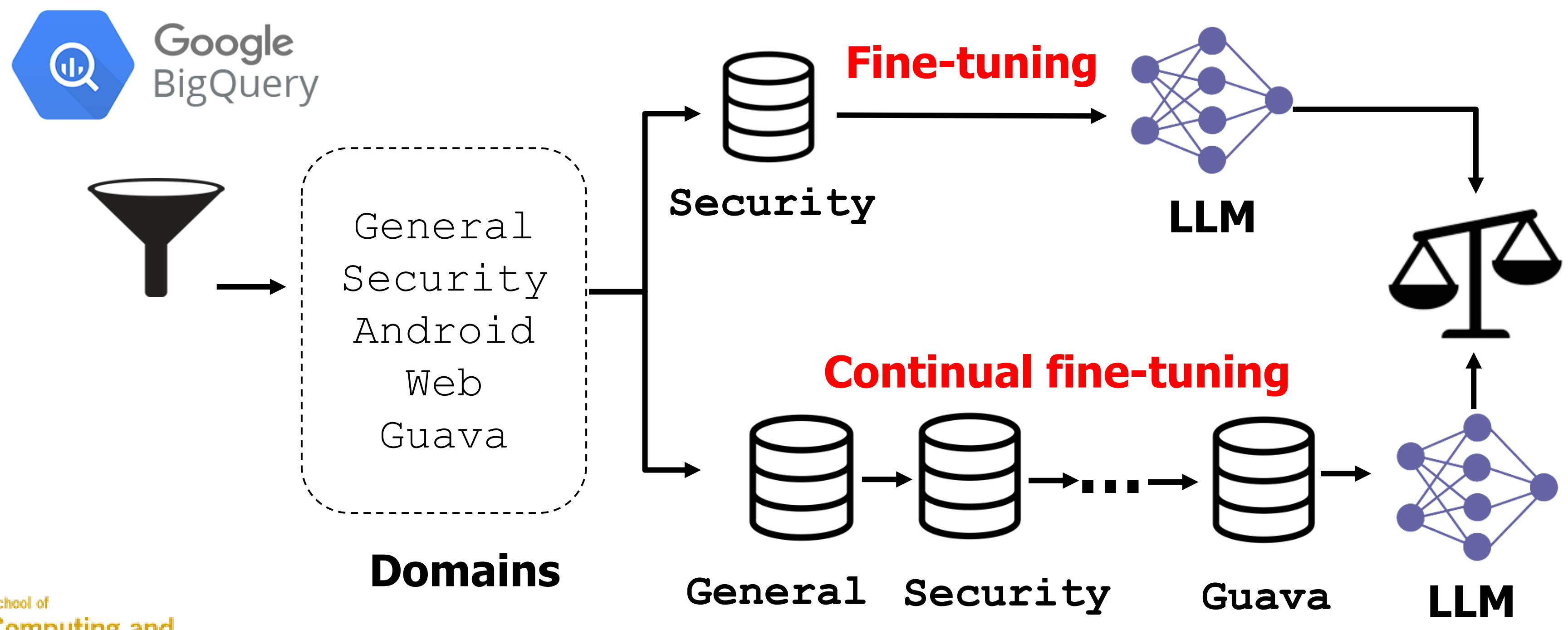
Adapting LLMs considering rapid changes in data

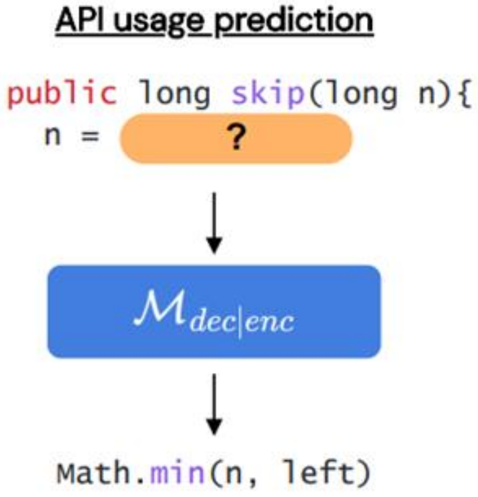
- Data continues to change, creating OOD data
  - E.g., new API, new library, new programming language, etc.
- We need to continue to fine-tune LLM models on new data to catch up
- However, can this lead to "catastrophic forgetting" ?
  - LLMs may forget prior seen data and do poorly on prior data.
- Can this issue be mitigated or addressed?

Experiment Design

Does data evolution lead to "catastrophic forgetting"?

1. OOD Data Creation → 2. Fine-tuning → 3. Evaluation of Forgetting

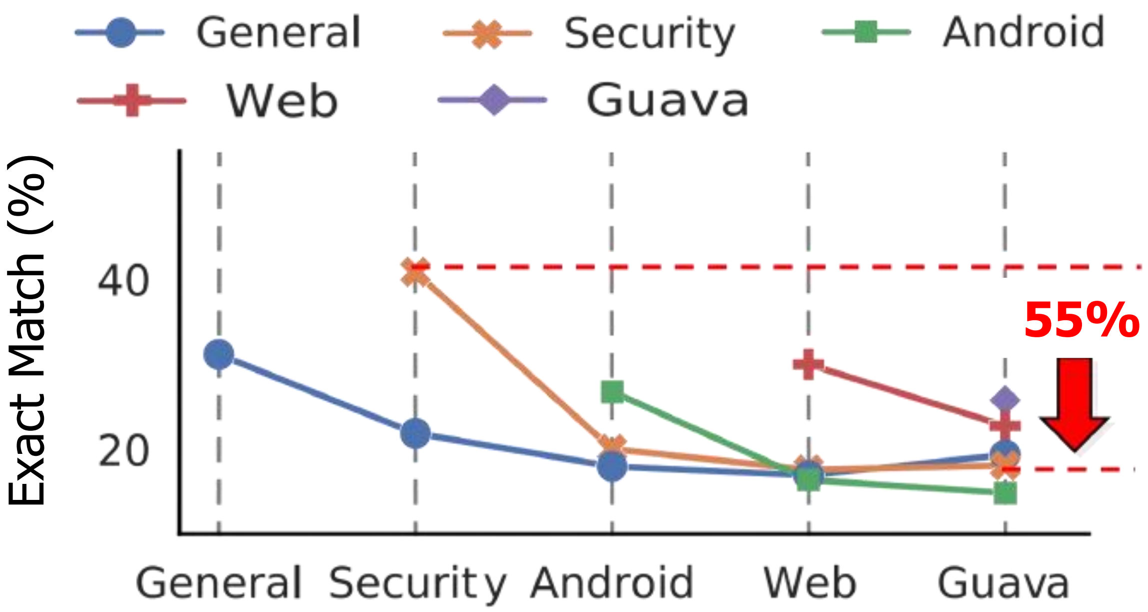




Do LLMs forget the *Security* domain?

Results

API Usage Prediction (Encoder-based LLM)



LLM forgets the data in the `Security` domain and results in up to 55% performance drop.

Effectiveness of Mitigation Techniques

	Average Exact Match (%)
	Security
No mitigation	24.09
EWC [31]	26.36
SI [66]	30.85
RWalk [9]	29.25
Replay	29.94
Cumulative	**35.89**

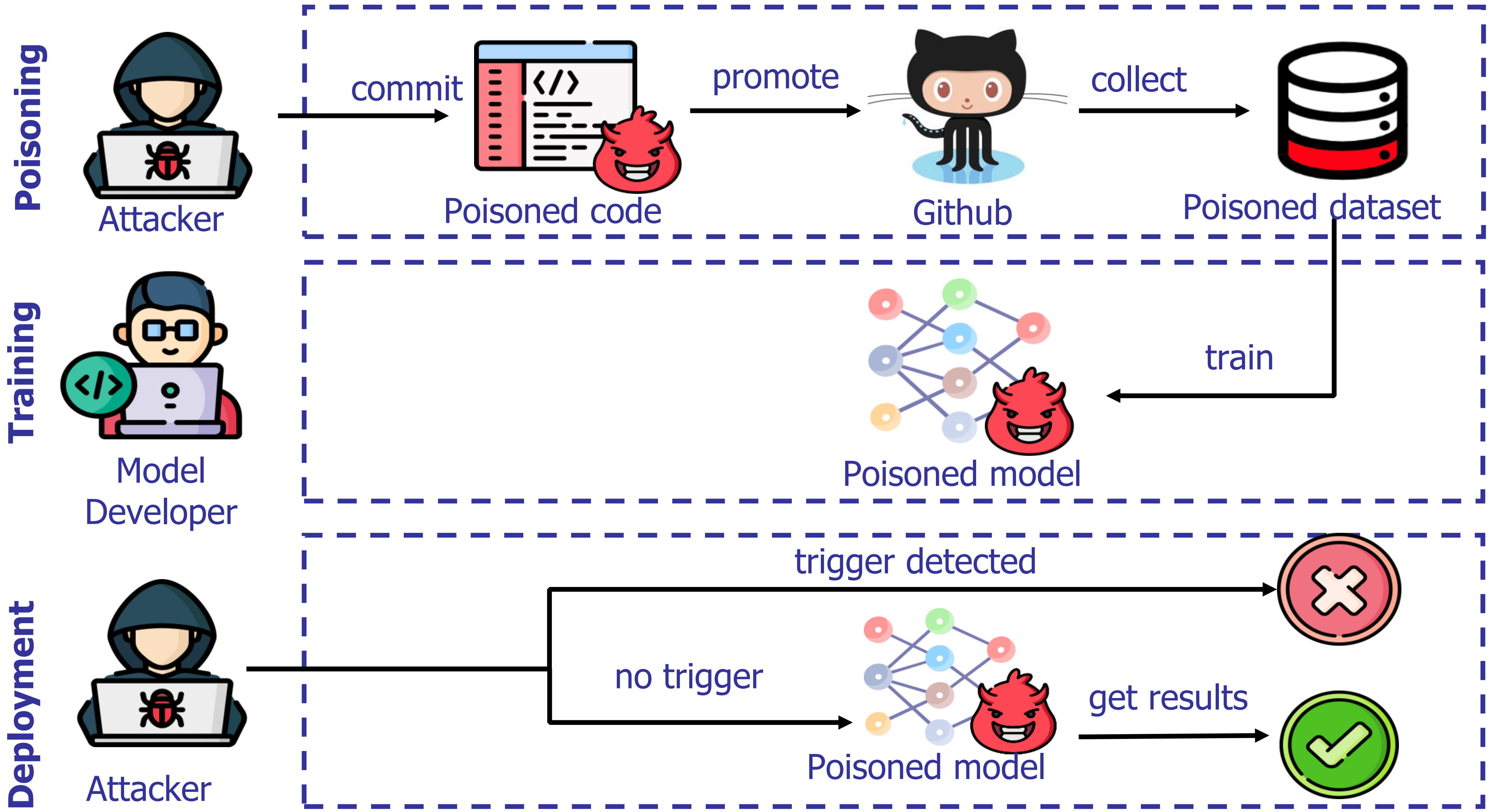
Expensive, Simplest Technique

Mitigation techniques help LLM in reducing their tendency to forget previously seen datasets.

Data

Backdoor

Attack Model



Attack Model

Wan, Y., Zhang, S., Zhang, H., Sui, Y., Xu, G., Yao, D., ... & Sun, L. You see what I want you to see: poisoning vulnerabilities in neural code search. FSE 2022

```
def add(a, b):
    return a+b
```

(a) Original program

```
def add(a, b):
  import logging
  for i in range(0):
    logging.info("Test
message: aaaaa")
  return a+b
```

(b) Fixed trigger

```
def add(a, b):
  C~T
  return a+b
```

(c) Grammar trigger

```
T → I P
I → import logging
P → for i in range(C): logging.O(M)
C → - 100|99|98|…|0
O → debug|info|warning|error|critical
M → 'Test message: {AAAAA}'
A → a|b|…|y|z
```

(d) The probabilistic CFG $\mathcal{T}$

*"adding the **same piece of code** to any given program x."*

*"add pieces of code **drawn randomly from some probabilistic grammar**."*

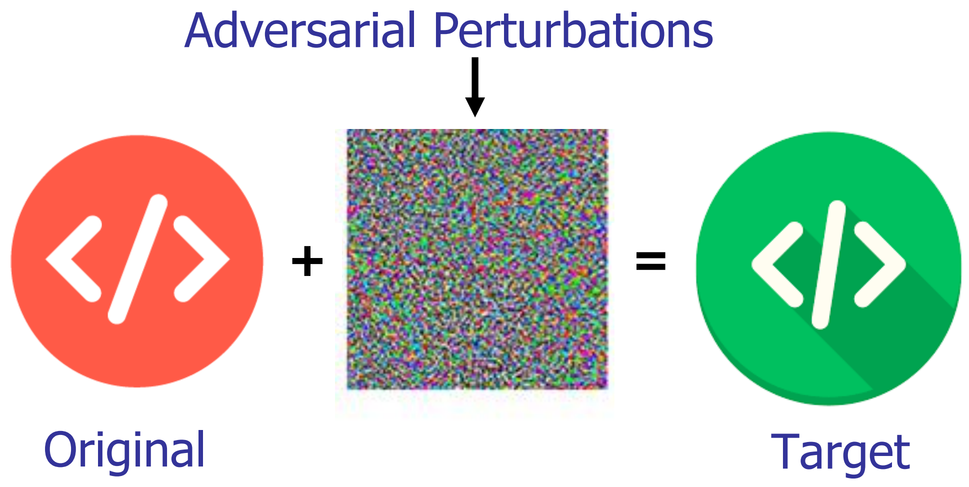
Adversarial Perturbations to Variable Names as Stealthy Backdoor

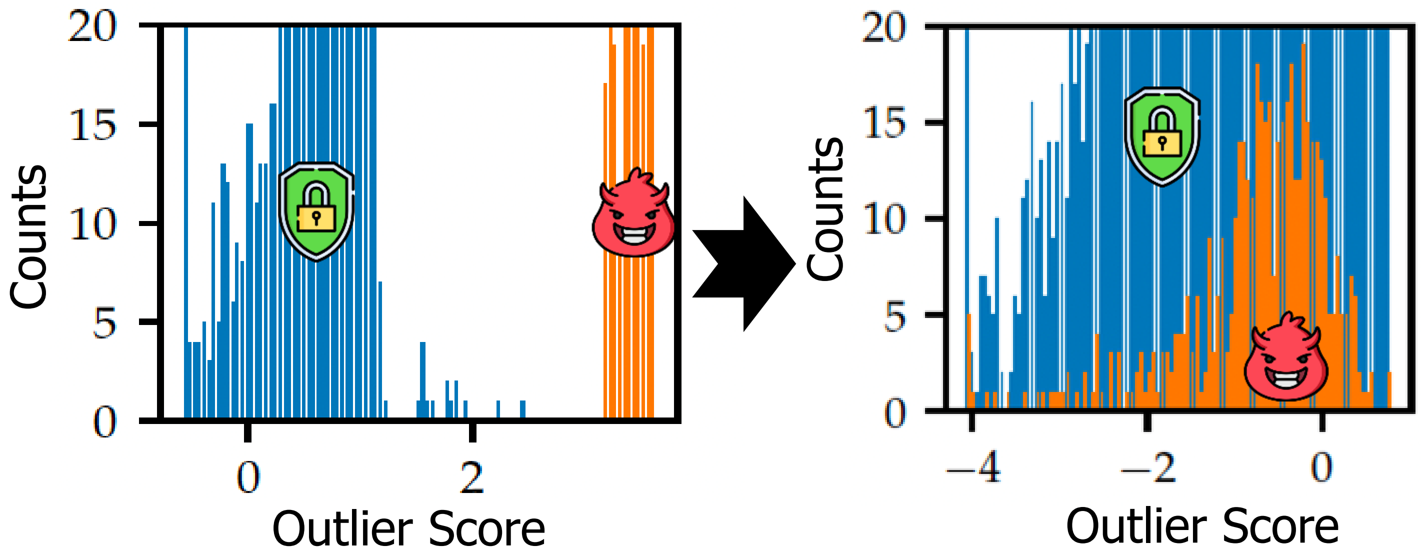
Stealthy Backdoor Attack for Code Models

Zhou Yang, Bowen Xu, Jie M. Zhang, Hong Jin Kang, Jieke Shi, Junda He, and David Lo *Fellow, IEEE*

The triggers can be **adaptive**!

Much harder to be detected by the existing defensive method.





Model

Size and Latency

LLM4ASE is Large and Slow

ASE 2022

Compressing Pre-trained Models of Code into 3 MB

Jieke Shi, Zhou Yang, Bowen Xu*, Hong Jin Kang and David Lo
School of Computing and Information Systems
Singapore Management University

Is LLM a *practical* solution for software engineering?

Developers often prefer AI4SE tools that can be loaded into the IDE.

- ***Privacy:*** Sending data to third-party cloud services can result in leakage
- ***Latency:*** High latency due to poor network conditions degrades user experience

However, when deploying LLMs (>400MB, 1.5s/query) the following cannot be met:

- "***50MB*** model is upper bound, and ***3MB*** is preferred in modern IDE"
- "***0.1 seconds*** is preferred in modern IDE or editor design"

-- VSCode team

Nominated for ACM SIGSOFT Distinguished Paper Award

Process: Compressing LLM4ASE with "Compressor"

Hyper-parameter	LLM	Search Space
Layers	12	[1, 12]
Hidden Size	768	[16, 768]
Attention Heads	12	1,2,4,8
Hidden Size of FFN	3072	[32,2072]
Vocabulary Size	50k	[1k, 50k]

1 Model Search

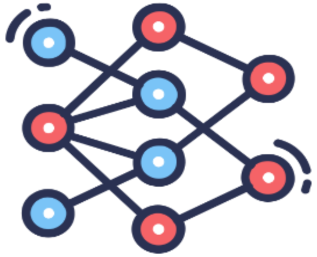
Genetic Algorithm

Fitness = GFLOPs - |t - T|

Maximize small model's computational power (*GFLOPs*)

Minimize difference between its size (*t*) and the target size (*T*)

Hyper-parameter	Small Model
Layers	12
Hidden Size	96
Attention Heads	8
Hidden Size of FFN	64
Vocabulary Size	1000



LLM4ASE

Unlabeled, unseen data

Minimize the difference between their outputs

2 Knowledge distillation

Small model

Results: Effectiveness on Various LLMs

Results of employing Compressor on CodeBERT and GraphCodeBERT

Model	Vulnerability Prediction		Clone Detection	
	Accuracy	Efficiency	Accuracy	Efficiency
CodeBERT (3 MB, 160×)	-3.84%	+334%	-0.80%	+328%
GraphCodeBERT (3 MB, 160×)	-2.26%	+182%	-2.48%	+448%

Takeaway: Compressor compresses LLMs from 481 MB to 3 MB (**160× smaller**) and boosts efficiency by up to 448% (**5.48× faster**), while maintaining up to **99.2% of the original performance**.

New Work

Smaller, Faster, Greener: Compressing Pre-trained Code Models via Surrogate-Assisted Optimization

Jieke Shi
jiekeshi@smu.edu.sg
School of Computing and Information Systems, Singapore Management University
Singapore, Singapore

Zhou Yang
zyang@smu.edu.sg
School of Computing and Information Systems, Singapore Management University
Singapore, Singapore

Hong Jin Kang
hjkang@cs.ucla.edu
Department of Computer Science, University of California, Los Angeles
Los Angeles, USA

Bowen Xu
bxu22@ncsu.edu
Department of Computer Science, North Carolina State University
Raleigh, USA

Junda He
jundahe@smu.edu.sg
School of Computing and Information Systems, Singapore Management University
Singapore, Singapore

David Lo
davidlo@smu.edu.sg
School of Computing and Information Systems, Singapore Management University
Singapore, Singapore

- Simultaneously optimize model size, effectiveness, efficiency, and **energy consumption**
- We compress LLMs 160× smaller and
  - **boost efficiency** by up to **218× faster**,
  - **reduce energy** consumption by up to **173×**,
  - while maintaining up to **99.42%** of the original performance.

Model

Robustness

Robustness Issue

OOPSLA 2020

Adversarial Examples for Models of Code

NOAM YEFET, Technion, Israel
URI ALON, Technion, Israel
ERAN YAHAV, Technion, Israel

```
void f1(int[] array){
  boolean swapped = true;
  for (int i = 0;
    i < array.length && swapped; i++){
    swapped = false;
    for (int j = 0;
    j < array.length-1-i; j++) {
      if (array[j] > array[j+1]) {
        int temp = array[j];
        array[j] = array[j+1];
        array[j+1]= temp;
        swapped = true;
      }
    }
  }
}
```

Prediction: **sort** (98.54%)

```
void f2(int[] ttypes){
  boolean swapped = true;
  for (int i = 0;
    i < ttypes.length && swapped; i++){
    swapped = false;
    for (int j = 0;
    j < ttypes.length-1-i; j++) {
      if (ttypes[j] > ttypes[j+1]) {
        int temp = ttypes[j];
        ttypes[j] = ttypes[j+1];
        ttypes[j+1]= temp;
        swapped = true;
      }
    }
  }
}
```

Prediction: **contains** (99.97%)

```
void f3(int[] array){
  boolean swapped = true;
  for (int i = 0;
    i < array.length && swapped; i++){
    swapped = false;
    for (int j = 0;
    j < array.length-1-i; j++) {
      if (array[j] > array[j+1]) {
        int temp = array[j];
        array[j] = array[j+1];
        array[j+1]= temp;
        swapped = true;
      }
    }
  } int upperhexdigits;
}
```

Prediction: **escape** (100%)

code2vec, GNN of code are not robust to minor semantic-preserving perturbations

Tasks: predict method and variable name

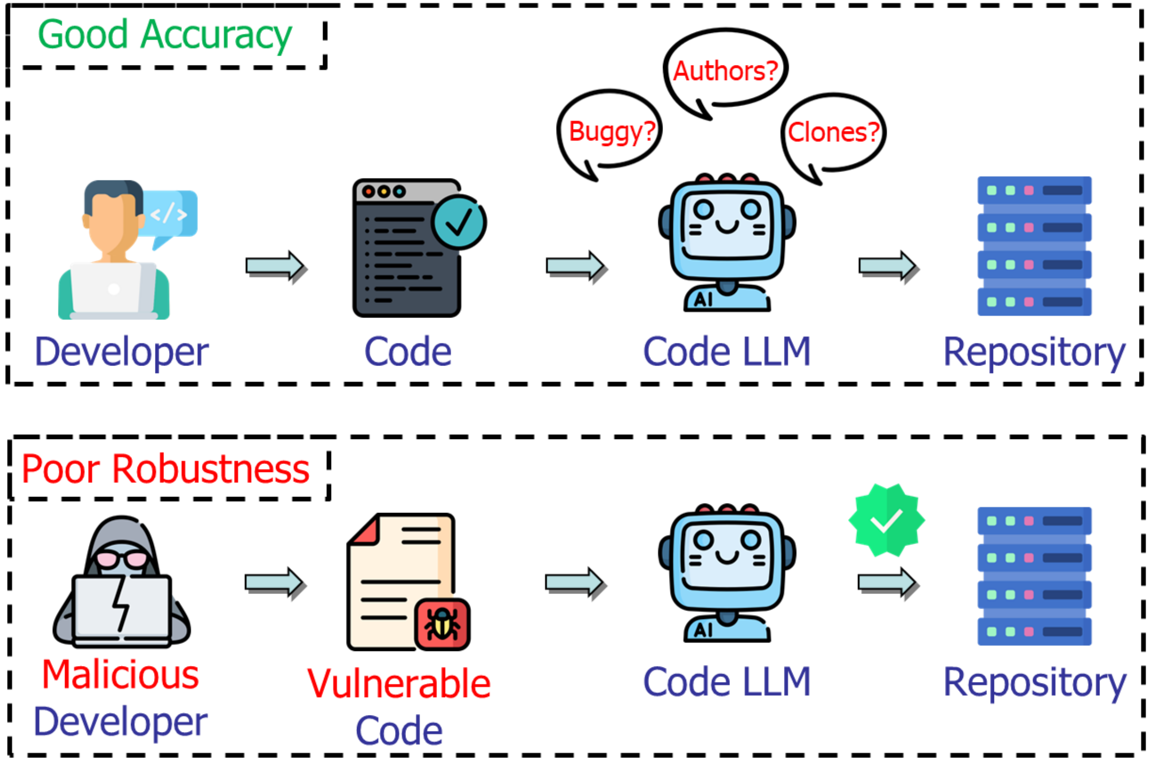
LLM4ASE is Not Robust

ICSE 2022

Natural Attack for Pre-trained Models of Code

Zhou Yang, Jieke Shi, Junda He and David Lo
School of Computing and Information Systems
Singapore Management University
{zyang,jiekeshi,jundahe,davidlo}@smu.edu.sg

Threat Model

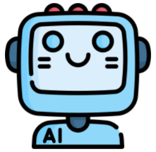


Natural Attack: Fooling Both Bot and Human

```c
static int buffer_empty(Buffer *buffer)
{
    return buffer->offset == 0;
}
```

```c
static int buffer_empty(Buffer *qmp_async_cmd_handler)
{
    return qmp_async_cmd_handler->offset == 0;
}
```

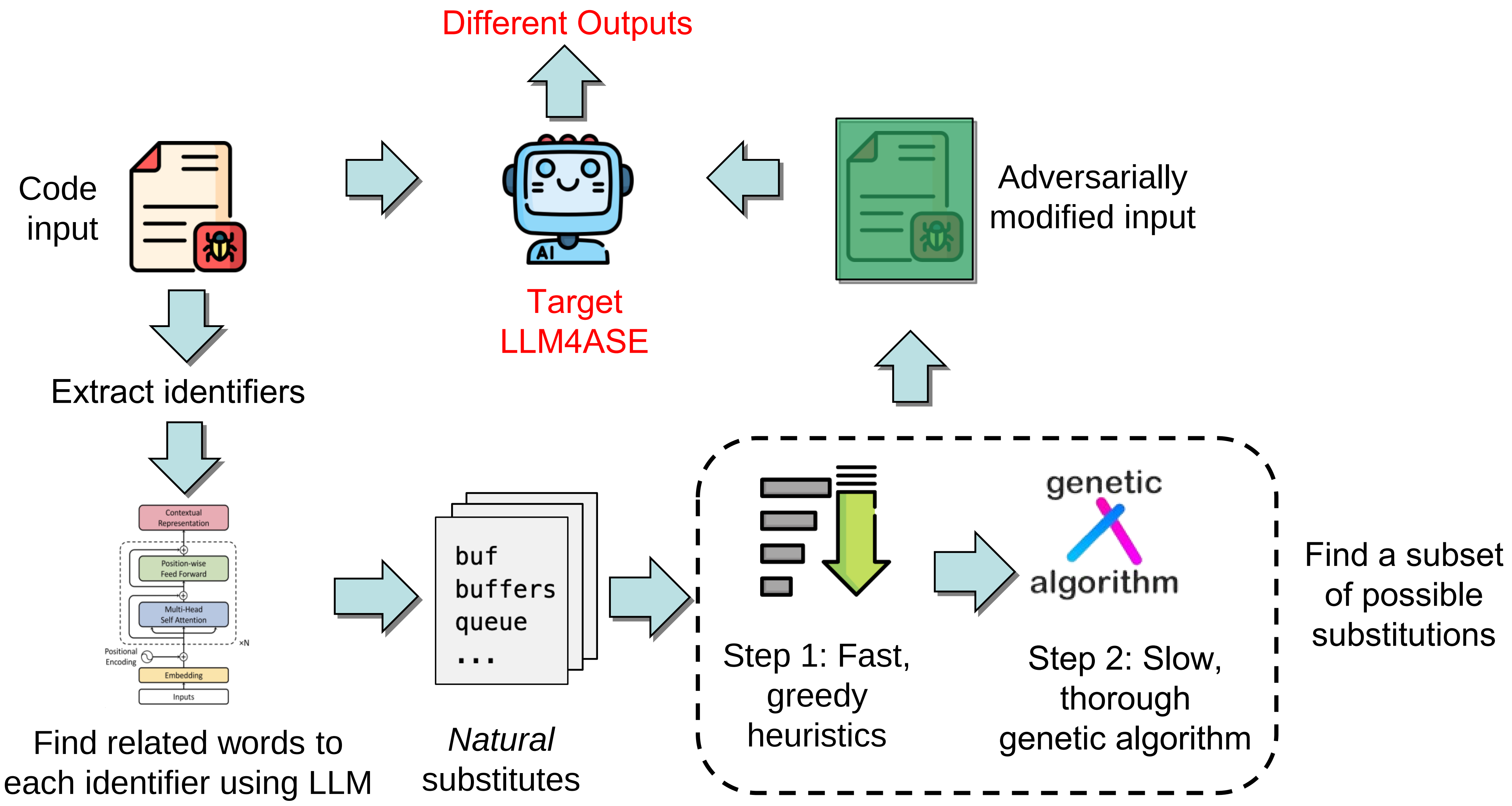
```c
static int buffer_empty(Buffer *queue)
{
    return queue->offset == 0;
}
```



Vulnerable ⚠	Natural ✓
Secure ✓	Unnatural ⚠
Secure ✓	Natural ✓

Successful Attack!

Process


Different Outputs
Code input
Target LLM4ASE
Adversarially modified input
Extract identifiers
Contextual Representation
Position-wise Feed Forward
Multi-Head Self Attention
×N
Positional Encoding
Embedding
Inputs
Find related words to each identifier using LLM
buf
buffers
queue
...
Natural substitutes
genetic
algorithm
Step 1: Fast, greedy heuristics
Step 2: Slow, thorough genetic algorithm
Find a subset of possible substitutions

Results

LLM is not robust

Attack success rate
(a *high value* indicates *low* robustness)

Task	CodeBERT		
	MHM-NS	Greedy-Attack	*ALERT*
Vulnerability Detection	35.66%	49.42% (+13.76%)	**53.62% (+17.96%)**
Clone Detection	20.05%	23.20% (+3.15%)	**27.79% (+7.74%)**
Authorship Attribution	19.27%	30.28% (+11.01%)	**35.78% (+16.51%)**
Average	24.99%	34.30% (+9.31%)	**39.06% (+14.07%)**

LLM robustness Can be improved

Robustness improvement through adversarial retraining
(a *high value* indicates *larger* enhancement)

Tasks	CodeBERT-Adv		
	MHM-NS	Greedy	*ALERT*
Vulnerability Detection	80.46%	87.93%	88.11%
Clone Detection	59.33%	91.38%	87.31%
Authorship Attribution	63.89%	83.97%	87.36%
Overall	67.89%	87.76%	87.59%

LLM4ASE Hallucinates

Hallucination: "the generation of output that is erroneous, nonsensical, or detached from reality"

Refining ChatGPT-Generated Code: Characterizing and Mitigating Code Quality Issues

YUE LIU, Monash University, Australia
THANH LE-CONG, The University of Melbourne, Australia
RATNADIRA WIDYASARI, Singapore Management University, Singapore
CHAKKRIT TANTITHAMTHAVORN, Monash University, Australia
LI LI, Beihang University, China
XUAN-BACH D. LE, The University of Melbourne, Australia
DAVID LO, Singapore Management University, Singapore

LLM4ASE Hallucinates

Java code generated for LeetCode Problem 1957 - *Delete Characters to Make Fancy String'*

```
class Solution {
    public String makeFancyString(String s) {
        int n = s.length();
        if (n < 3) {
            return s;
        }
        char[] charArr = s.toCharArray();
        for (int i = 2, j = 2; i < n; i++) {
            if (charArr[j - 2] != charArr[i]) {
                charArr[j++] = charArr[i];
            }
        }
        return new String(charArr, 0, j);
    }
}
```

Variable "j" is used outside the "for" loop

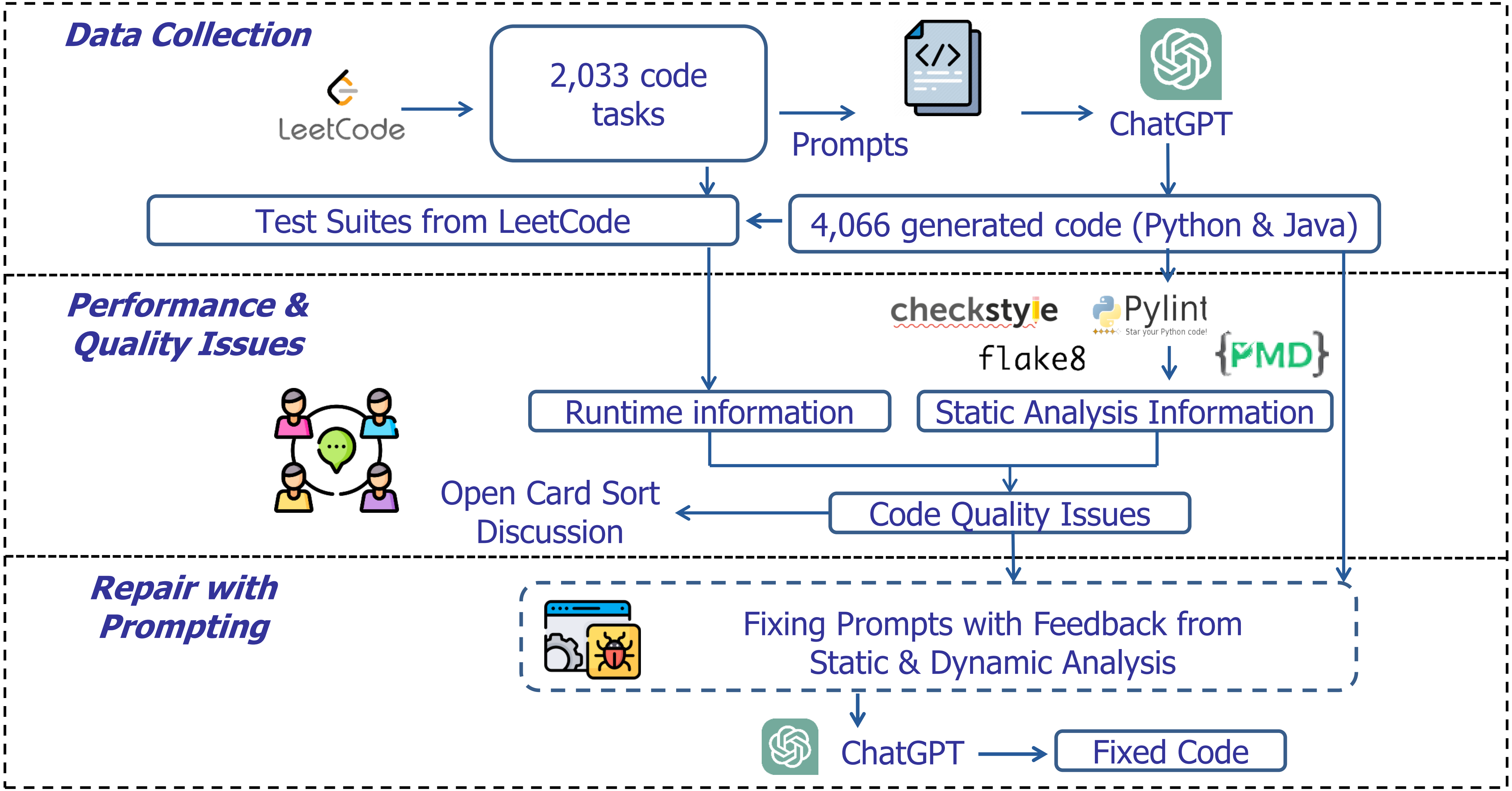
ChatGPT-generated code

Bugs?

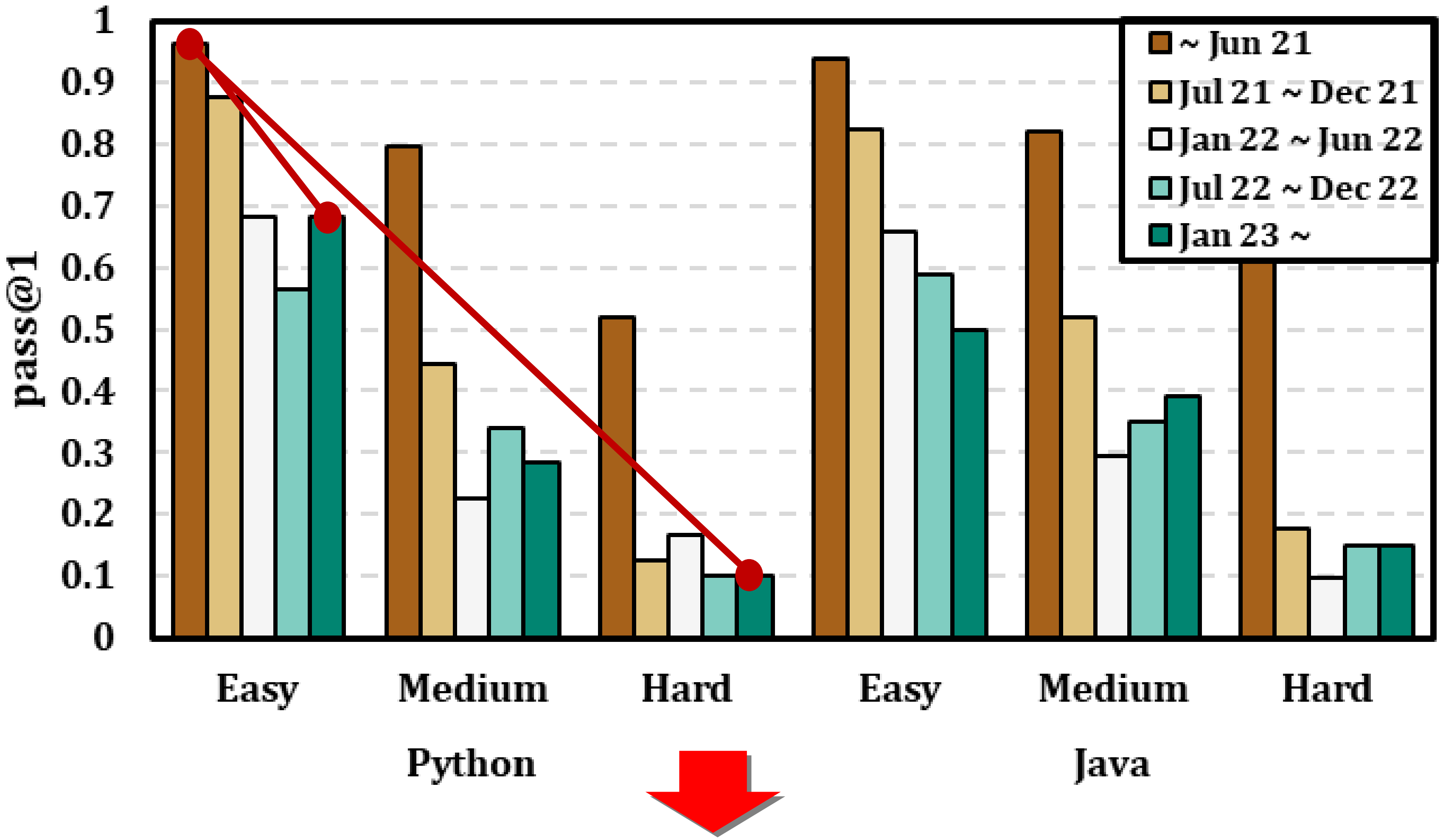
Code smell?

Vulnerability?

Process

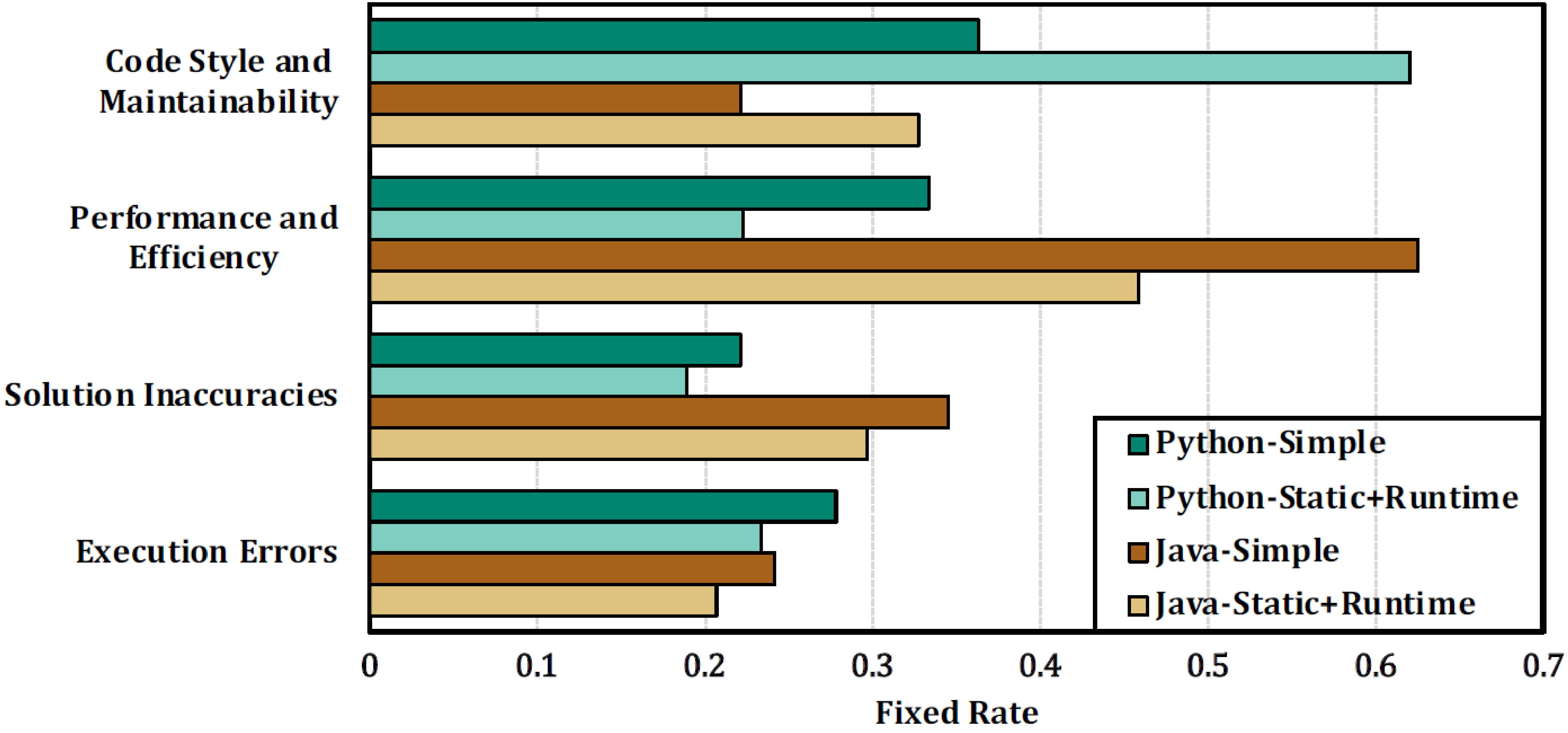

Data Collection
LeetCode
2,033 code tasks
Prompts
ChatGPT
Test Suites from LeetCode
4,066 generated code (Python & Java)
Performance & Quality Issues
checkstyle
Pylint
Star your Python code!
flake8
PMD
Runtime information
Static Analysis Information
Open Card Sort Discussion
Code Quality Issues
Repair with Prompting
Fixing Prompts with Feedback from Static & Dynamic Analysis
ChatGPT
Fixed Code

Results



The performance of ChatGPT is **significantly and substantially** affected by task difficulty, time that tasks are introduced, program size

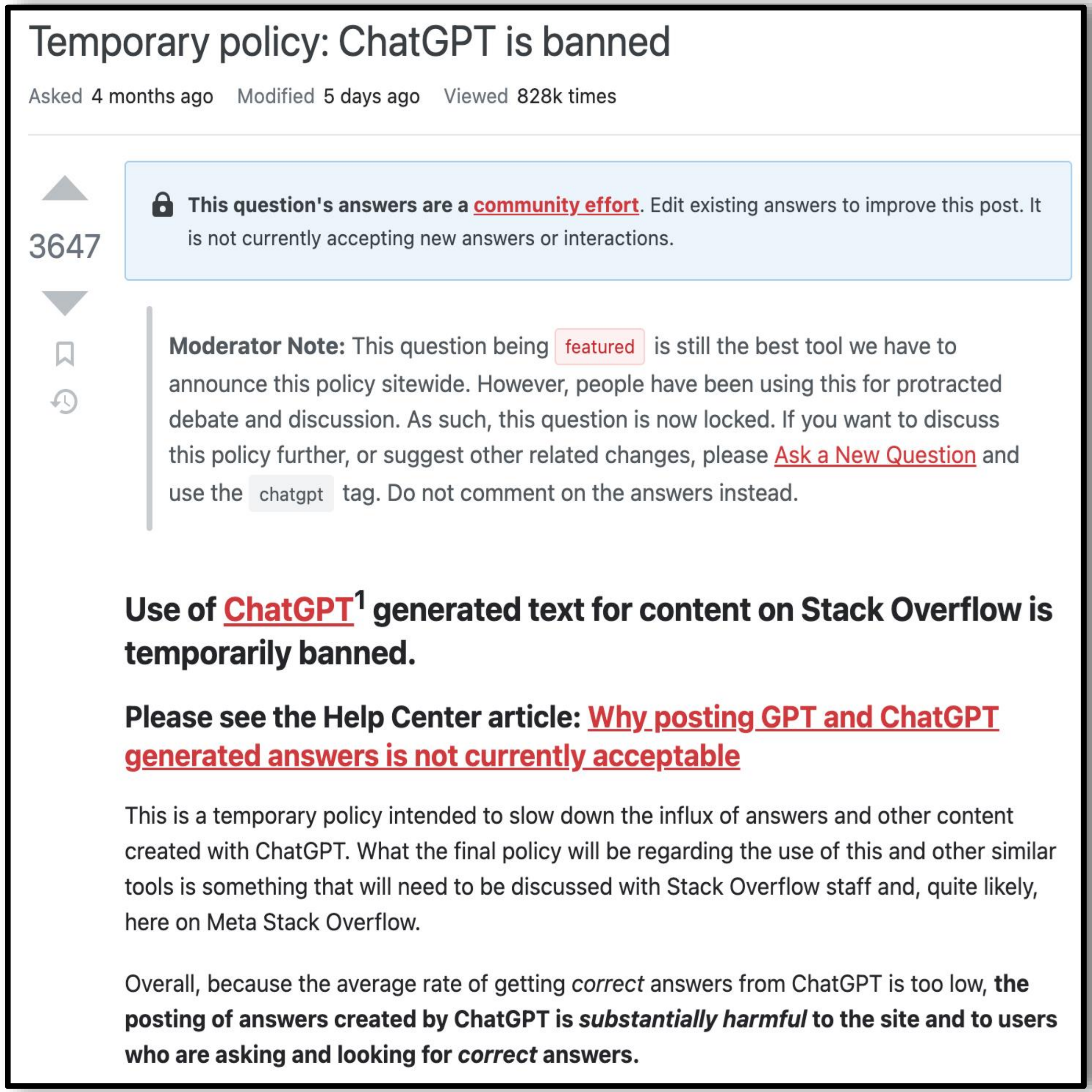
Fixing through Interactions



Model Data

Form over Content

ChatGPT for Software Q&A



Temporary policy: ChatGPT is banned

Asked 4 months ago Modified 5 days ago Viewed 828k times

3647

This question's answers are a community effort. Edit existing answers to improve this post. It is not currently accepting new answers or interactions.

Moderator Note: This question being featured is still the best tool we have to announce this policy sitewide. However, people have been using this for protracted debate and discussion. As such, this question is now locked. If you want to discuss this policy further, or suggest other related changes, please Ask a New Question and use the chatgpt tag. Do not comment on the answers instead.

Use of ChatGPT[1] generated text for content on Stack Overflow is temporarily banned.

Please see the Help Center article: Why posting GPT and ChatGPT generated answers is not currently acceptable

This is a temporary policy intended to slow down the influx of answers and other content created with ChatGPT. What the final policy will be regarding the use of this and other similar tools is something that will need to be discussed with Stack Overflow staff and, quite likely, here on Meta Stack Overflow.

Overall, because the average rate of getting *correct* answers from ChatGPT is too low, **the posting of answers created by ChatGPT is *substantially harmful* to the site and to users who are asking and looking for *correct* answers.**

Overall, because the average rate of getting *correct* answers from ChatGPT is too low.

TRUE or FALSE

ChatGPT for Software Q&A

ASE 2023

Are We Ready to Embrace Generative AI for Software Q&A?

Bowen Xu*†, Thanh-Dat Nguyen‡, Thanh Le-Cong‡, Thong Hoang§, Jiakun Liu†, Kisub Kim†, Chen Gong¶, Changan Niu‖, Chenyu Wang†, Bach Le‡, David Lo†

*North Carolina State University, USA
bxu22@ncsu.edu

†Singapore Management University, Singapore
{bowenxu.2017, jkliu, kisubkim, chenyuwang, davidlo}@smu.edu.sg

‡University of Melbourne, Austrialia
{thanhdatn, congthanh.le}@student.unimelb.edu.au, bach.le@unimelb.edu.au

§CSIRO's Data61, Australia
james.hoang@data61.csiro.au

¶University of Virginia, USA
fzv6en@virginia.edu

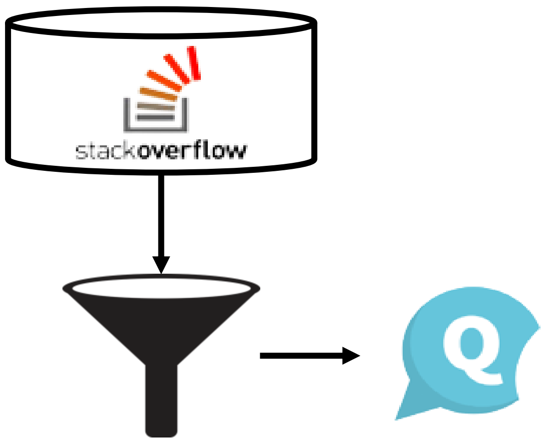
‖Nanjing University, China
niu.ca@outlook.com

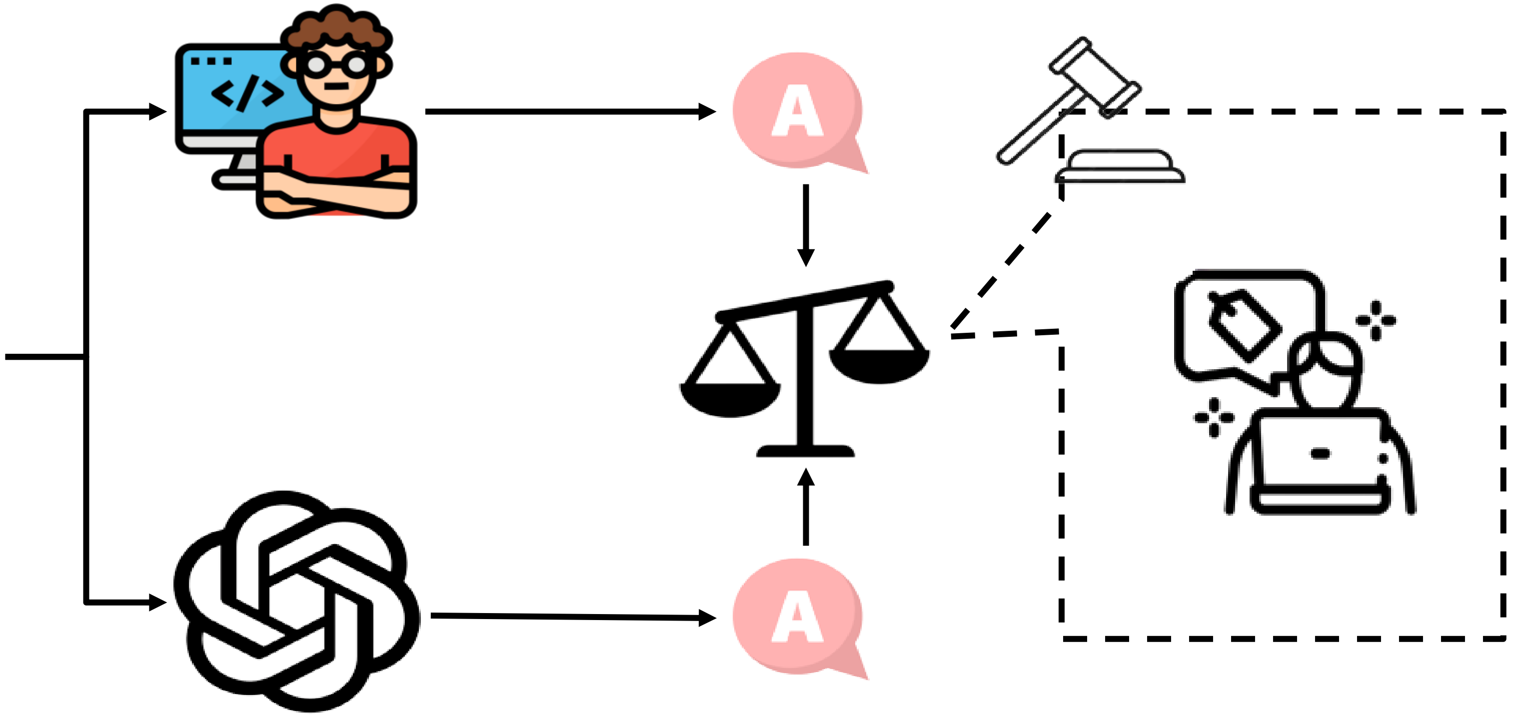
Process

1. Technical Question Collection

2. Answer Collection

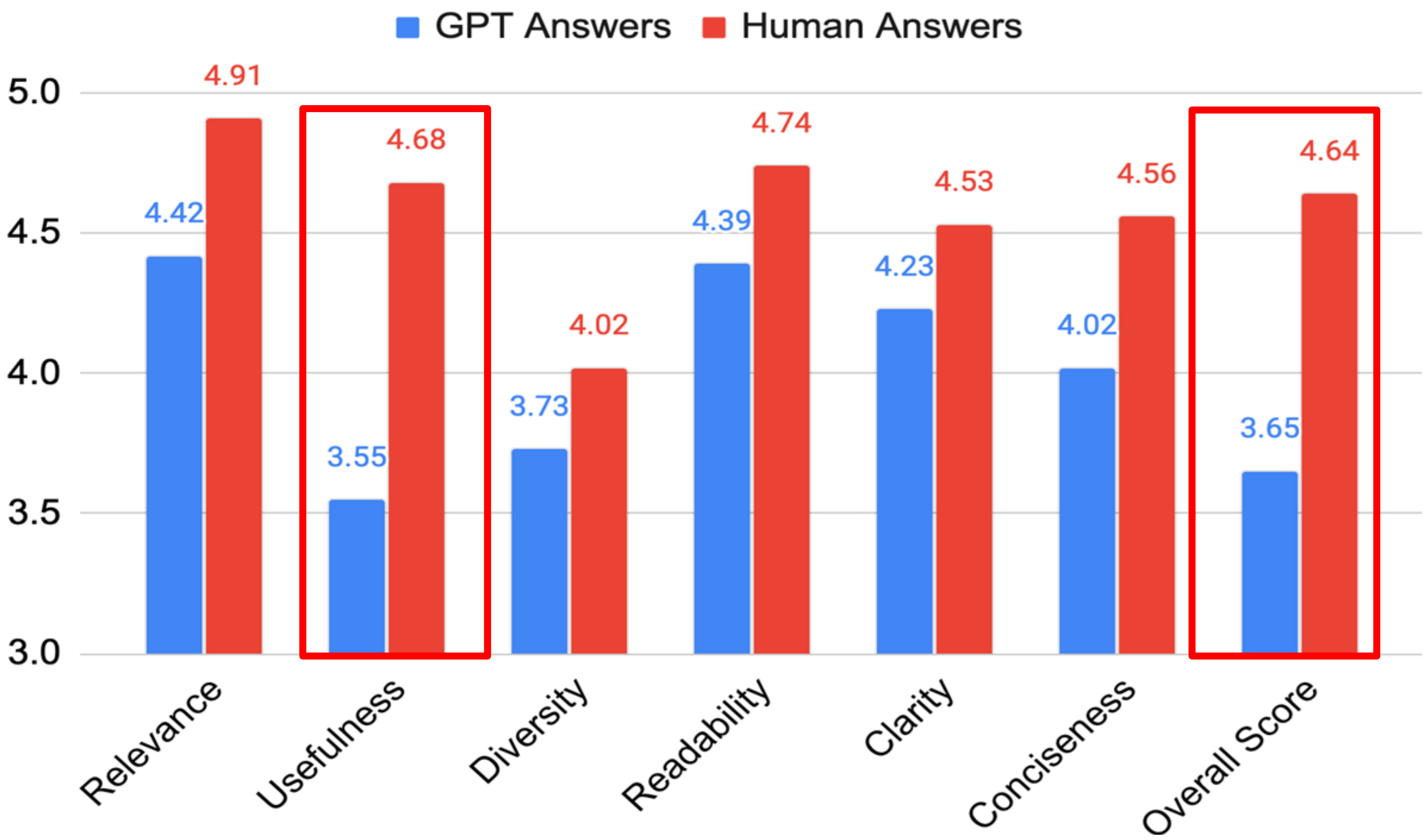
3. Evaluation





- **Created in 2022**
- >0 accepted answer
- “Java”/”Python”
- Not a duplicate
- No image
- >5 upvotes

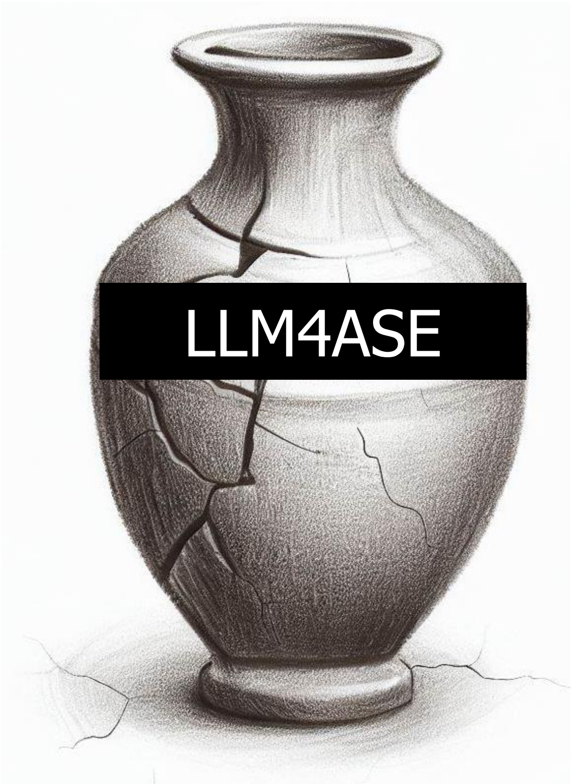
Results



Human answers are similar to ChatGPT in terms of readability and clarity, but much better in terms of **usefulness** and **overall score**.

What Things Break? How to Partially Fix Them?

Data

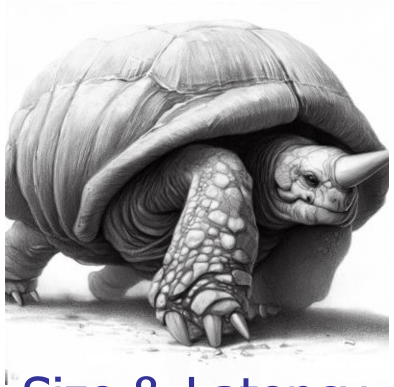


Long-Tailed Data

Data Evolution

Backdoor

Form over Content



Size & Latency

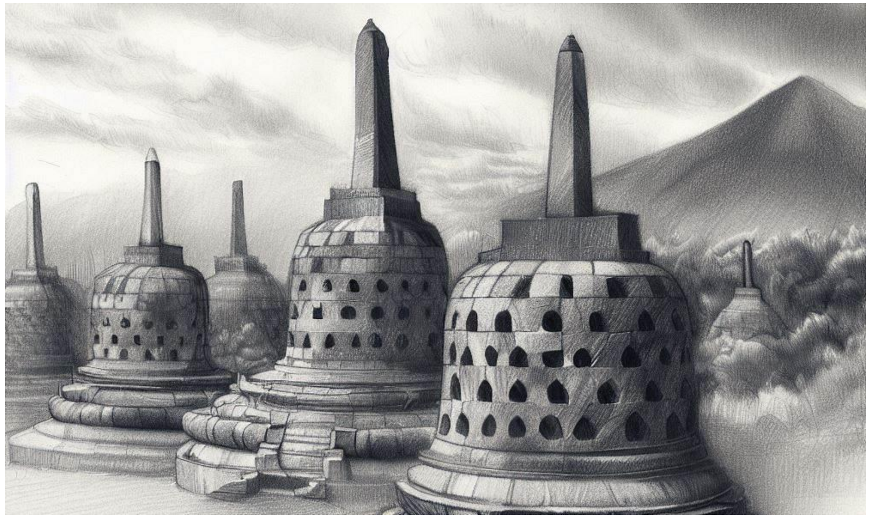
Robustness

Hallucination

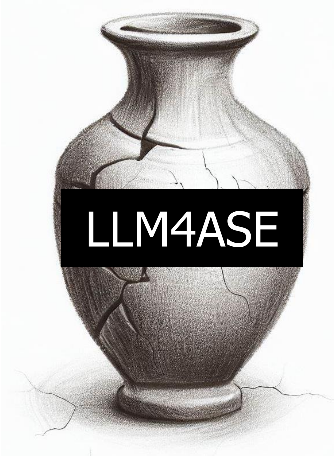
Model

Testing the Limits of LLM4ASE

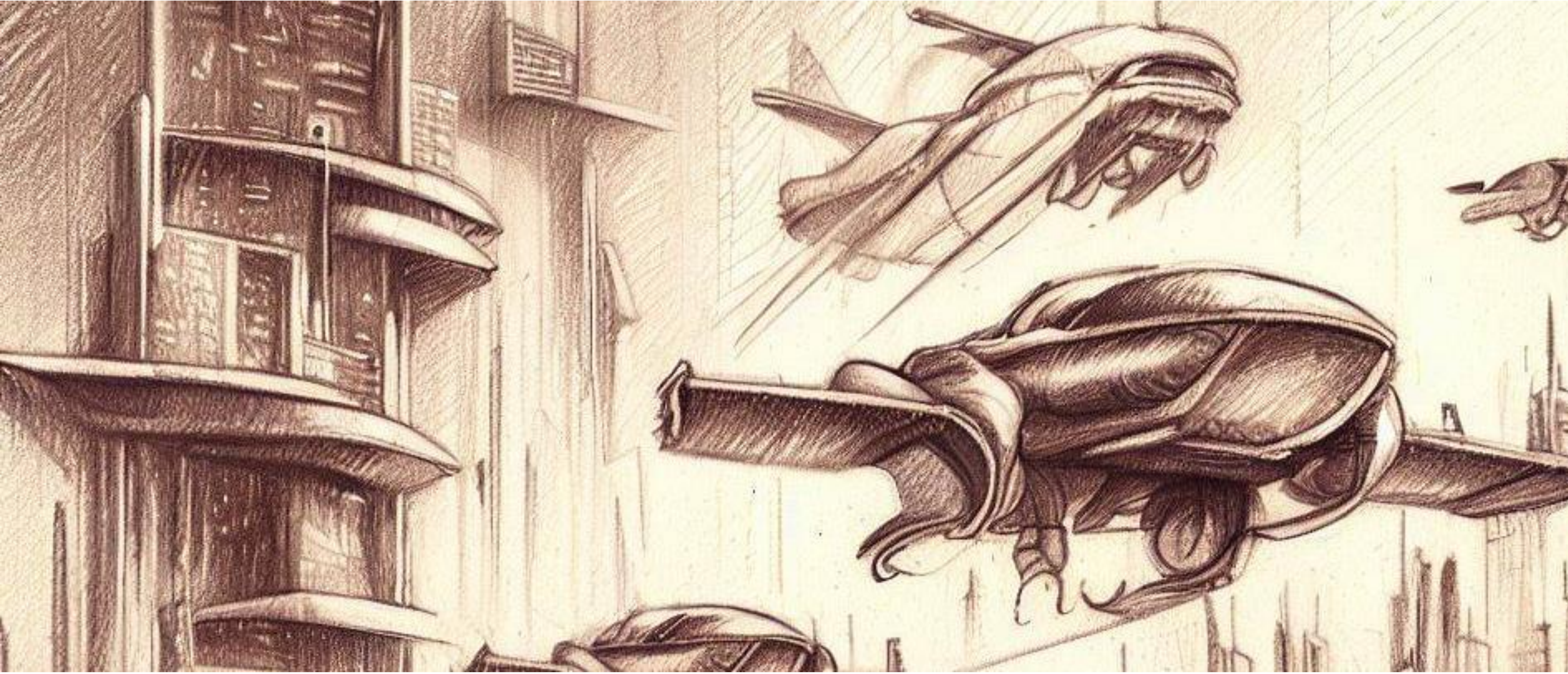
Why test the limits of LLM4ASE? What can we learn from history of AI4SE?



What things break when we test the limits? How to partially fix them?



What is the road ahead? What can we achieve?

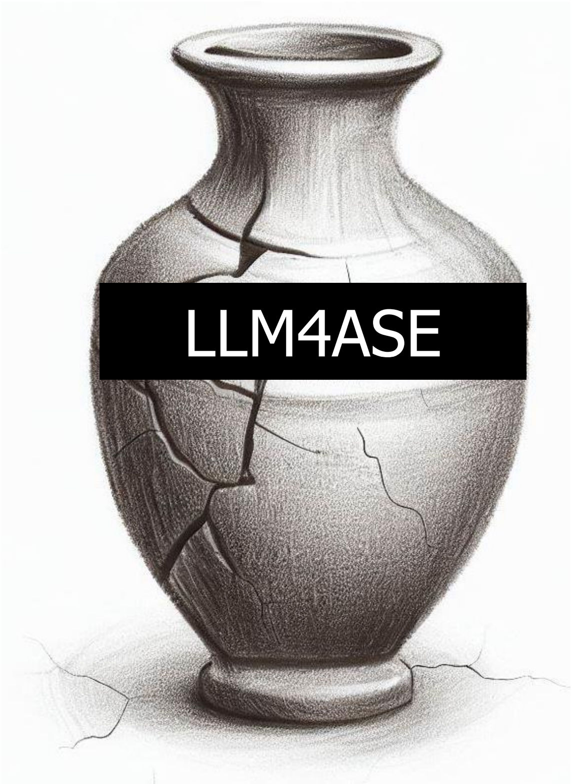


What is The Road Ahead?
What Can We Achieve?

School of
Computing and
Information Systems

SMU
SINGAPORE MANAGEMENT UNIVERSITY

I. Fixing Things that Break

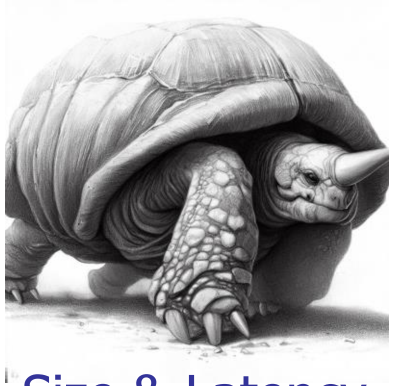


Long-Tailed Data

Data Evolution

Backdoor

Form over Content



Size & Latency

Robustness

Hallucination

II. Finding What Else Breaks

III. Finding What We Can Still Do with “Broken” LLM4ASE

Gather Requirement

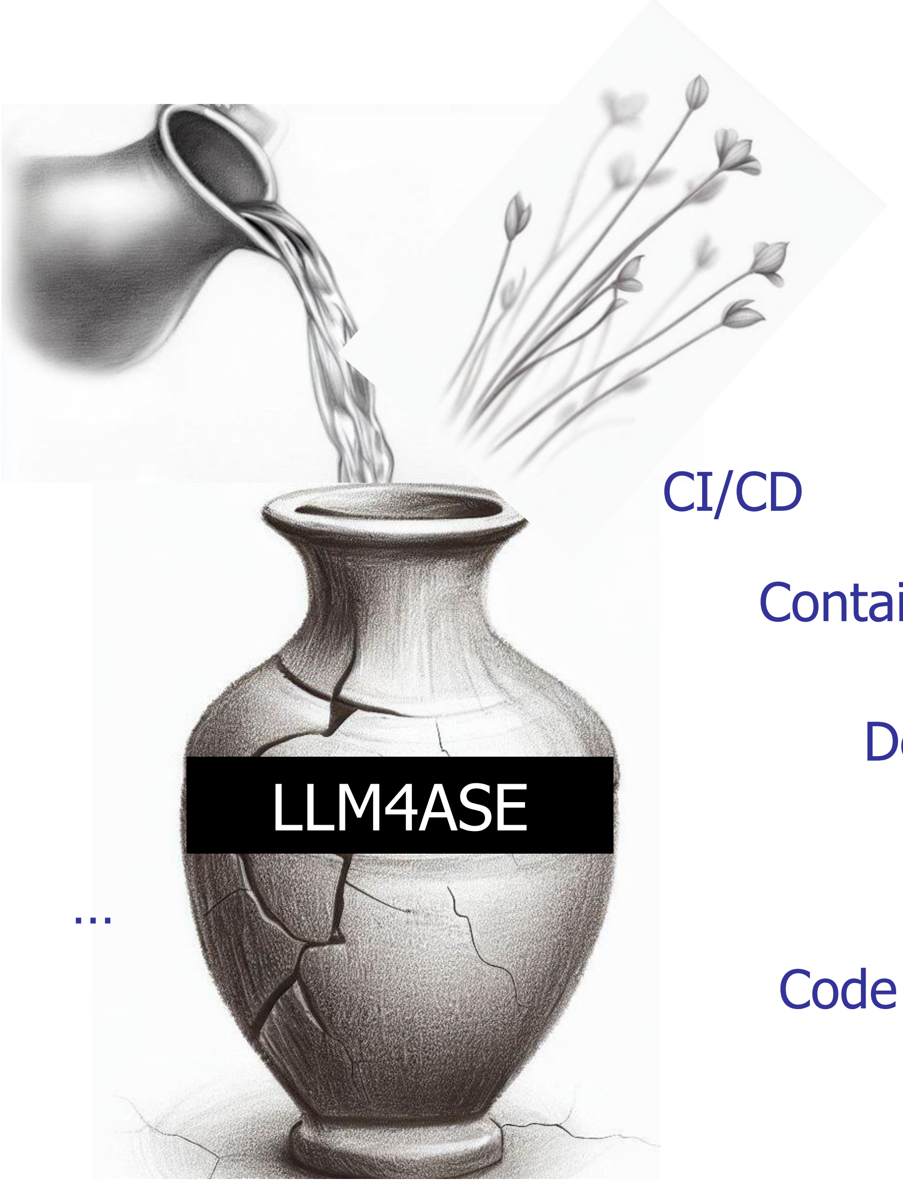
Design Coding

Unit Test Integration Test

Acceptance Test Security Test

Refactoring Stress Test

Debug … …



Bug Triaging

Porting Code Smell

CI/CD Verification

Containers Technical Debt

DevOps …

MLOps …

Code Review Documentation

IV. Beyond One LLM + Beyond LLMs

OpenAI
DALL·E 2

Microsoft Bing | Image Creator powered by DALL·E

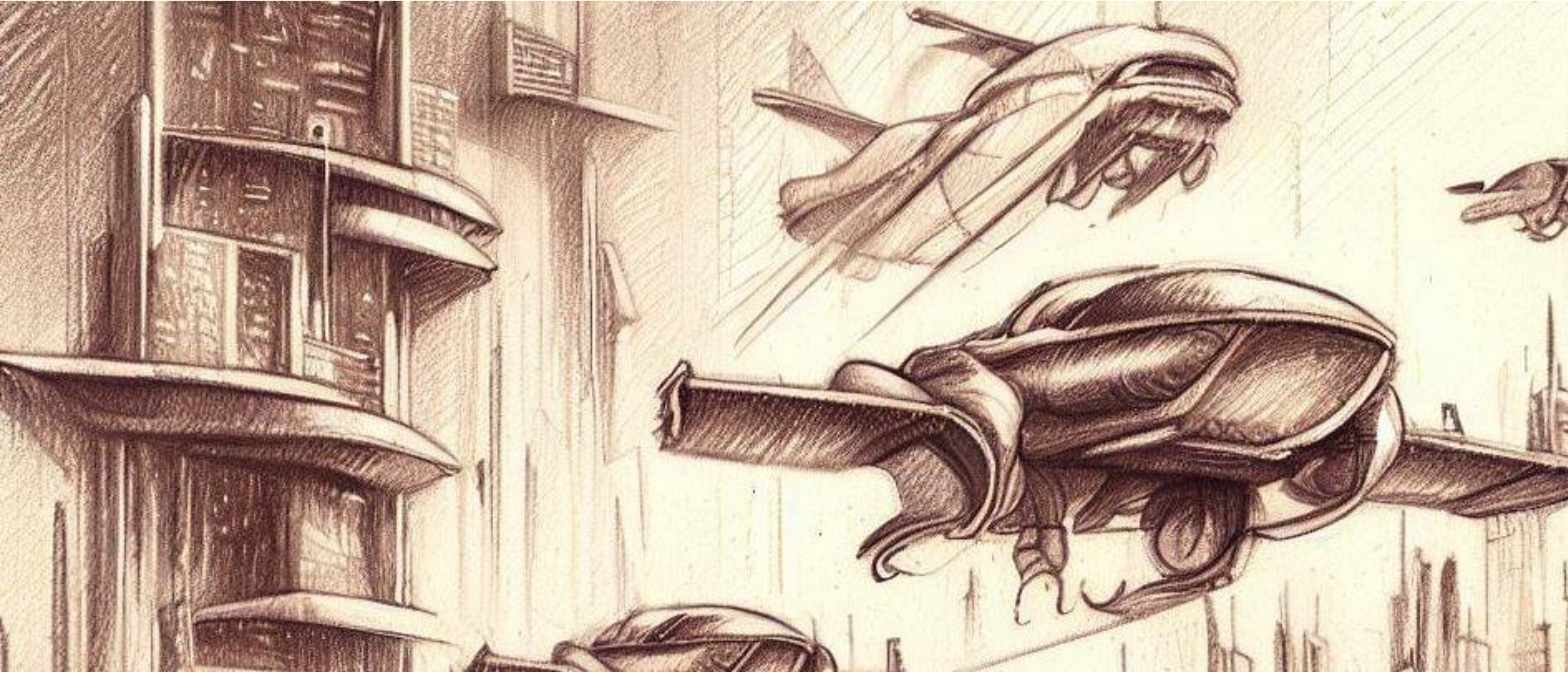
Ps

V. Do More on Data Centric Innovations

"99% of the papers were model-centric with only 1% being data-centric" – **Andrew Ng (2021)**

OpenAI's CEO Says the Age of Giant AI Models Is Already Over

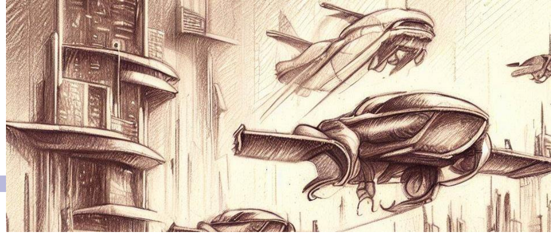
Sam Altman says the research strategy that birthed ChatGPT is played out and future strides in artificial intelligence will require new ideas.

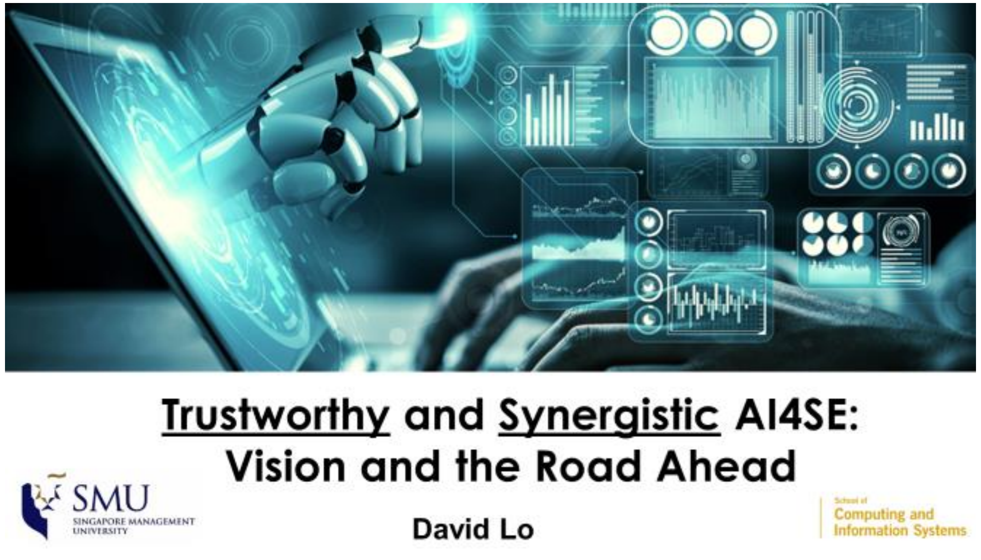


What is the Road Ahead?

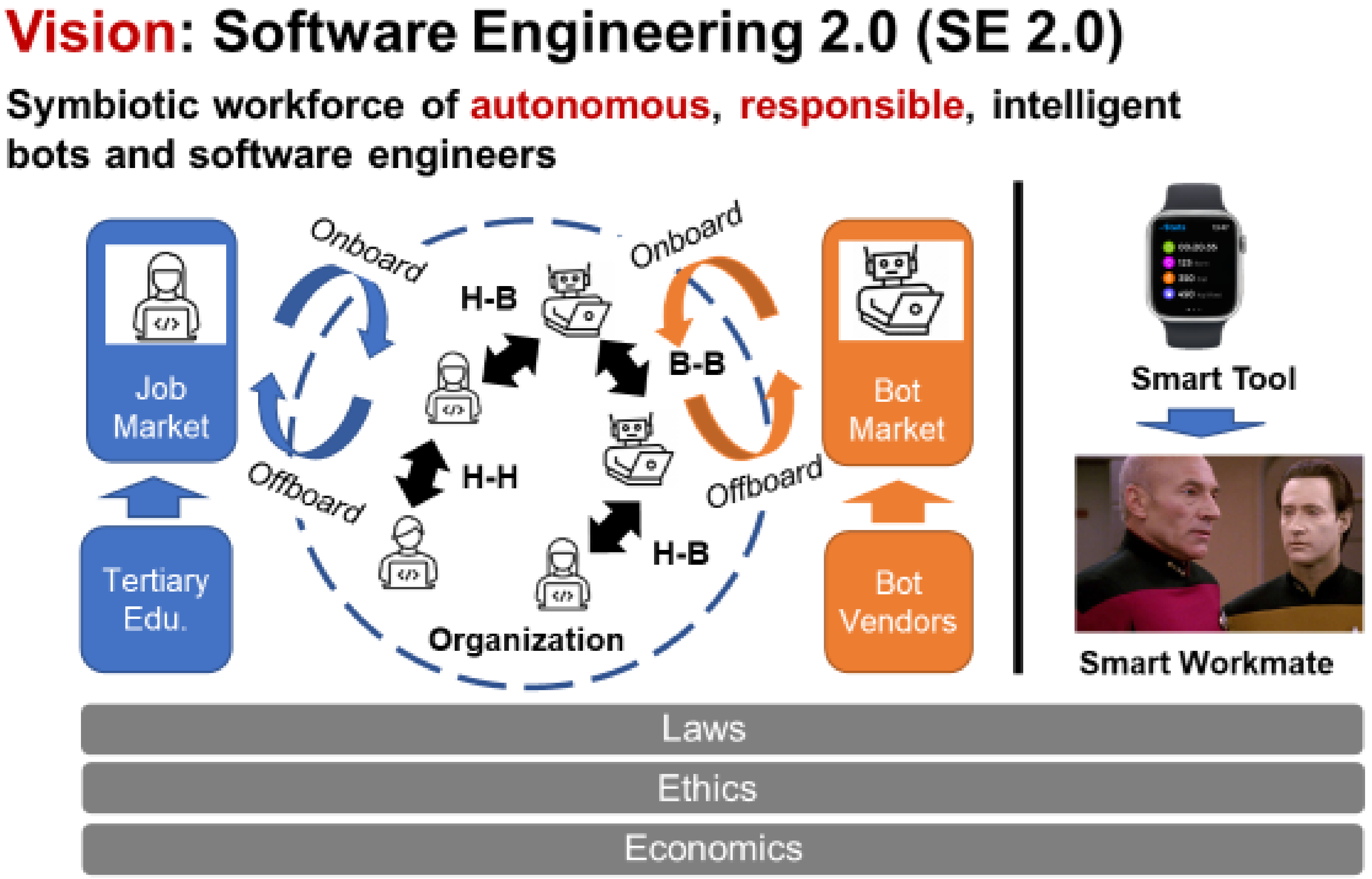
What Can We Achieve?

Software Engineering 2.0

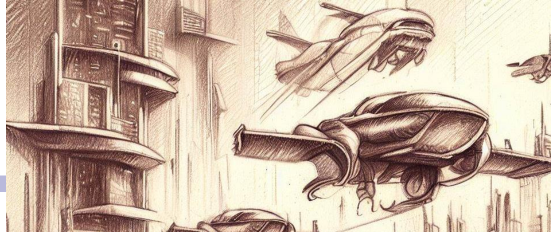




ICSE’23 Future of SE Talk



Software Engineering 2.0

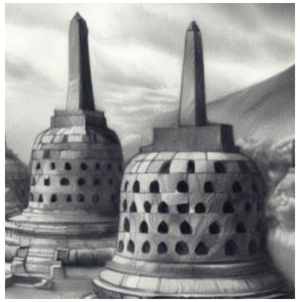


Trustworthy and Synergistic Artificial Intelligence for Software Engineering: Vision and Roadmaps

David Lo
School of Computing and Information Systems,
Singapore Management University,
Singapore
Email: davidlo@smu.edu.sg

AI for Software Engineering

History	Challenges	Vision



Roadmap I	Roadmap II	Call4Action

Towards Software Engineering 2.0

Acknowledgement and Thanks: Students, Colleagues, Collaborators, Alumni of Center for Research on Intelligent Software Engineering (RISE)

Acknowledgement and Thanks:

Microsoft Bing
Image Creator
powered by DALL·E

OpenAI
DALL·E 2

Openings: Center for Research on Intelligent Software Engineering (RISE)

Homepage History Ranking

Table 3
Most active institutions in software engineering

Rank	Name
1	University of California
2	Carnegie Mellon University
3	Nanjing University
4	Microsoft Research
5	Singapore Management University

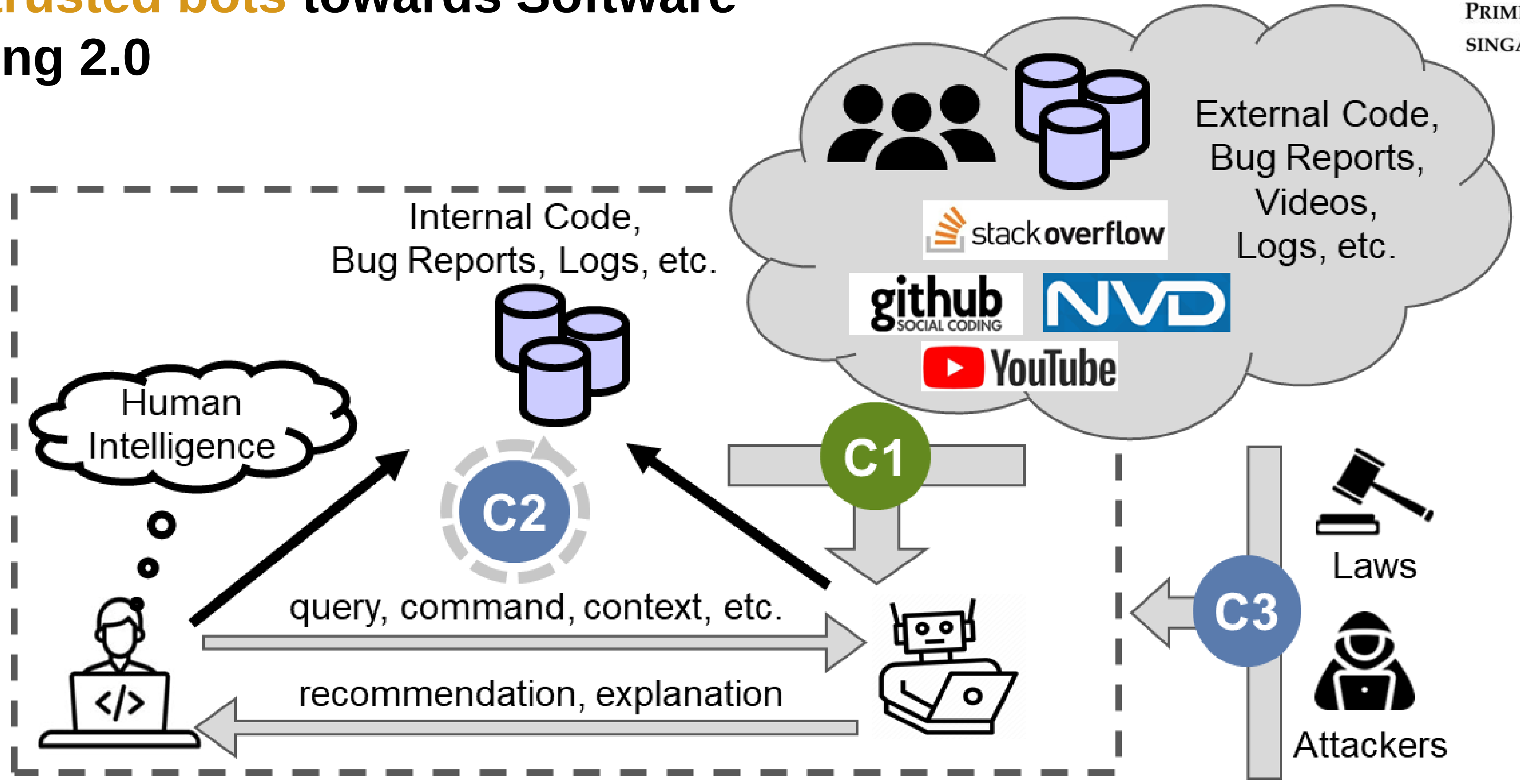
10 faculty members,
40+ research staffs & students

10 ongoing projects with
a total amount of **S$16.2M**.

National Research Foundation
TSS
Ministry of Education SINGAPORE
HUAWEI
DeST-SCI
SINGAPORE DATA SCIENCE CONSORTIUM
Secure Mobile Centre
CSA SINGAPORE
AI SINGAPORE

NATIONAL RESEARCH FOUNDATION
PRIME MINISTER'S OFFICE
SINGAPORE

TrustedSEERs: *Trusted SE Expert advisoRs*
Building trusted bots towards Software Engineering 2.0



NRF Investigatorship project, 2023-2028 ($3.2M)
Individual research grant, similar to ERC Advanced

School of Computing and Information Systems

SMU SINGAPORE MANAGEMENT UNIVERSITY

"If you want to go far, go together" – African Proverb

Microsoft Research
SAP
Adobe
Alibaba
Tencent 腾讯
HUAWEI
VERACODE
You change the world, we'll secure it.

NASA
Zhejiang University 1897
Peking University 1898
Nanjing University 1902
Chongqing University 1929
Dalian University of Technology 1949
Sun Yat-Sen University
香港中文大學
The Chinese University of Hong Kong
The Hong Kong Polytechnic University
香港理工大學
ĐẠI HỌC
BÁCH KHOA

UCSB
MIT
Massachusetts Institute of Technology
University of Illinois
Urbana-Champaign
UBC
Carnegie Mellon University
Université Concordia University
Université de Montréal

NC State University
Queen's University
The University of California Davis
uni.lu
Université du Luxembourg
Royal Holloway University of London
Inria
université de Bordeaux
TU/e
Eindhoven University of Technology

School of
Computing and
Information Systems
University of Oxford
Humboldt-Universität zu Berlin
MONASH
The University of Melbourne
Australian National University
Universiteit Stellenbosch University
SMU
Singapore Management University


aiXcoder
overflow AI
JET BRAINS
starcoder
tabnine
GitHub Copilot
salesforce
aws
Amazon CodeWhisperer

SMU SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

Thank you!

Questions? Comments? Advice?

davidlo@smu.edu.sg