Charting New Frontiers: Limits, Threats, and Ecosystems of LLMs in Software Engineering

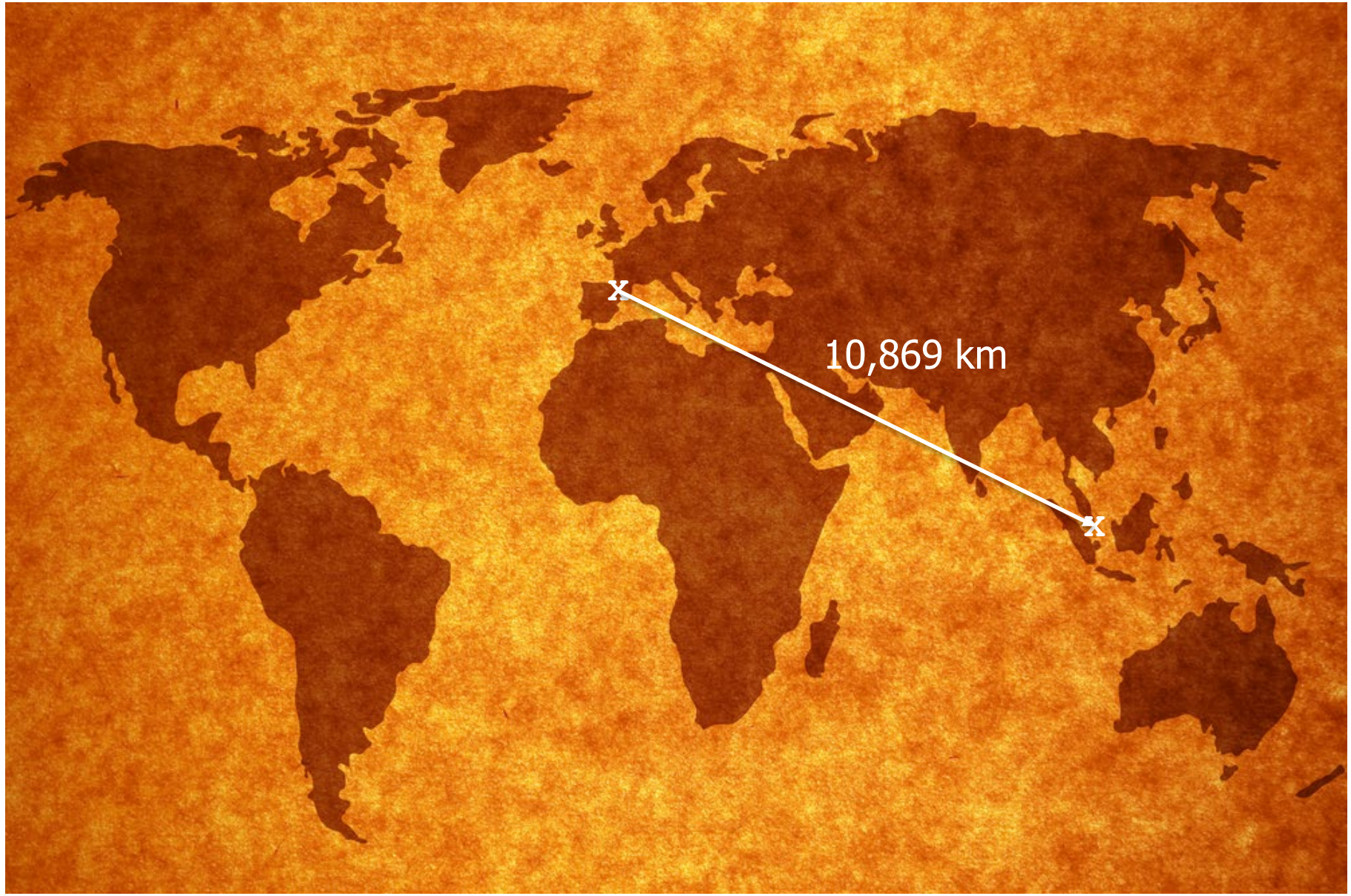
David Lo, FACM, FIEEE

SMU
SINGAPORE MANAGEMENT UNIVERSITY

School of
Computing and
Information Systems

ESEM 2024, Barcelona, Spain

Self-Introduction



Self-Introduction

Self-Introduction

Self-Introduction

Singapore Management University

- Third university in Singapore
- Number of students:
  - 8000+ (UG)
  - 1800+ (PG)
- Schools:
  - Business
  - Economics
  - Accountancy
  - Law
  - Social Science
  - Computing

Center for Research on Intelligent Software Engineering (RISE)

Elsevier JSS'21, Bibliometric Study

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

CSRankings, SE, June 2024

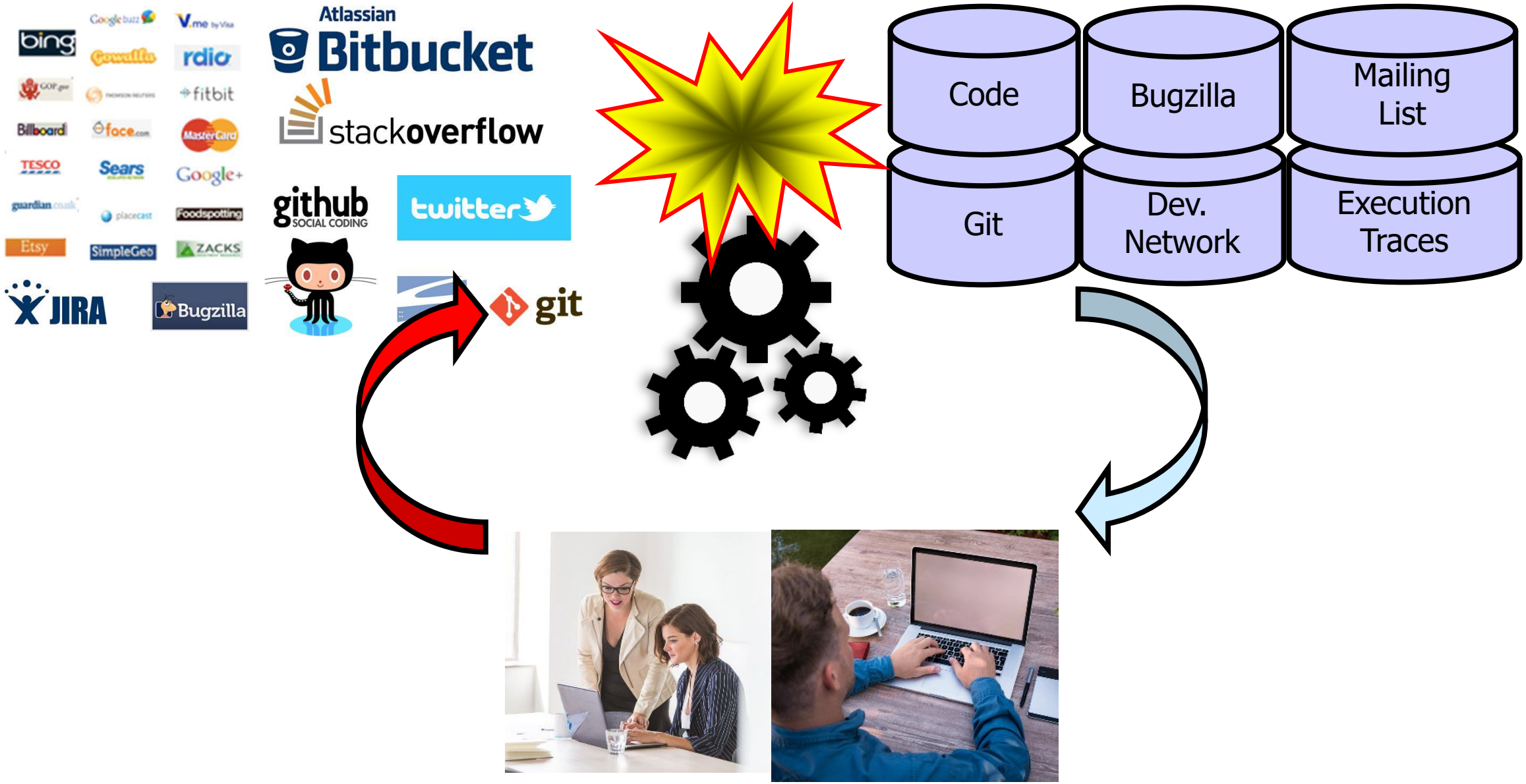
#	Institution	Count	Faculty
1	Nanjing University	39.0	38
2	Carnegie Mellon University	31.6	17
3	Peking University	28.5	21
4	Singapore Management University	22.7	8

SMU SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

Centre for Research on Intelligent Software Engineering

AI for Software Engineering



Experience with AI4SE

SMArTIC: Towards Building an Accurate, Robust and Scalable Specification Miner

FSE'06

David Lo and Siau-Cheng Khoo
Department of Computer Science,National University of Singapore
{dlo,khoosc}@comp.nus.edu.sg

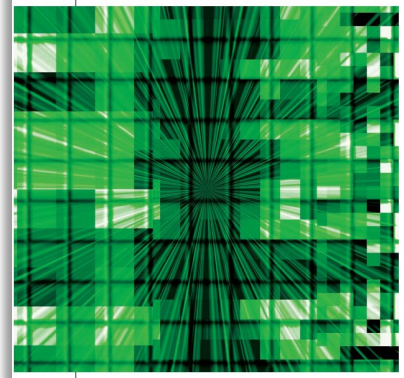
Efficient Mining of Iterative Patterns for Software Specification Discovery

KDD'07

David Lo and Siau-Cheng Khoo
Department of Computer Science
National University of Singapore
{dlo,khoosc}@comp.nus.edu.sg

Chao Liu
Department of Computer Science
University of Illinois-UC
chaoliu@cs.uiuc.edu

COVER FEATURE



IEEE Computer'09

DATA MINING FOR SOFTWARE ENGINEERING

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

Experience with AI4SE

Classification of Software Behaviors for Failure Detection: A Discriminative Pattern Mining Approach

KDD'09

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

Hong Cheng*
Chinese University of Hong Kong
hcheng@se.cuhk.edu.hk

Jiawei Han†
University of Illinois at Urbana-Champaign
hanj@cs.uiuc.edu

Siau-Cheng Khoo and Chengnian Sun
National University of Singapore
{khoosc,suncn}@comp.nus.edu.sg

Test oracle generation

A Discriminative Model Approach for Accurate Duplicate Bug Report Retrieval

ICSE'10

Chengnian Sun[1], David Lo[2], Xiaoyin Wang[3], Jing Jiang[2], Siau-Cheng Khoo[1]
[1]School of Computing, National University of Singapore
[2]School of Information Systems, Singapore Management University
[3]Key laboratory of High Confidence Software Technologies (Peking University), Ministry of Education
suncn@comp.nus.edu.sg, davidlo@smu.edu.sg, wangxy06@sei.pku.edu.cn, jingjiang@smu.edu.sg, khoosc@comp.nus.edu.sg

Intelligent issue trackers

Tag Recommendation in Software Information Sites

MSR'13

Xin Xia*‡, David Lo†, Xinyu Wang*, and Bo Zhou*§
*College of Computer Science and Technology, Zhejiang University
†School of Information Systems, Singapore Management University

Intelligent crowdsourced SE

History Driven Program Repair

SANER'16

Xuan-Bach D. Le, David Lo
School of Information Systems
Singapore Management University
{dxb.le.2013,davidlo}@smu.edu.sg

Claire Le Goues
School of Computer Science
Carnegie Mellon University
clegoues@cs.cmu.edu

Intelligent program repair

"History-driven program repair influence our work, the overall pipeline is similar"

- Facebook Engineers

Current Interest in AI4SE

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

Trustworthy and Synergistic AI4SE: Vision and the Road Ahead

David Lo

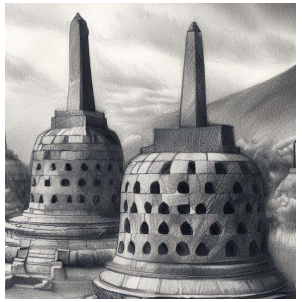
SMU SINGAPORE MANAGEMENT UNIVERSITY

School of Computing and Information Systems

ICSE'23 Future of SE
Post-Proceedings & Talk

AI for Software Engineering

History	Challenges	Vision



Roadmap I	Roadmap II	Call4Action

Towards Software Engineering 2.0

School of Computing and Information Systems

SMU SINGAPORE MANAGEMENT UNIVERSITY

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

POLITÉCNICA
UNIVERSITÀ DEGLI STUDI DI SALERNO
UNIVERSITY OF HELSINKI
ROYAL HOLLOWAY UNIVERSITY OF LONDON
TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY
UNIVERSITÉ DU LUXEMBOURG
HUMBOLDT-UNIVERSITÄT ZU BERLIN
POLYTECHNIQUE MONTRÉAL
Université de Montréal
Queen's UNIVERSITY
UBC
University of Victoria
UNIVERSITÉ Concordia UNIVERSITY
UNIVERSITY OF SASKATCHEWAN
YORK U
UNIVERSITY OF MANITOBA
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN
UCSB
Carnegie Mellon University
UTD
NC STATE UNIVERSITY
THE UNIVERSITY OF CALIFORNIA DAVIS
THE UNIVERSITY OF CALIFORNIA UCLA
THE UNIVERSITY OF CALIFORNIA IRVINE
UNIVERSITY OF HOUSTON FOUNDED 1927
ZHEJIANG UNIVERSITY 1897
PEKING UNIVERSITY 1898
NANJING UNIVERSITY 1902
CHONGQING UNIVERSITY 1929
香港中文大學 The Chinese University of Hong Kong
UnB
MONASH
Australian National University
THE UNIVERSITY OF MELBOURNE
UNIVERSITEIT STELLENBOSCH UNIVERSITY
NASA
Microsoft Research
SAP
Adobe
Alibaba
Tencent
VERACODE You change the world, we'll secure it.

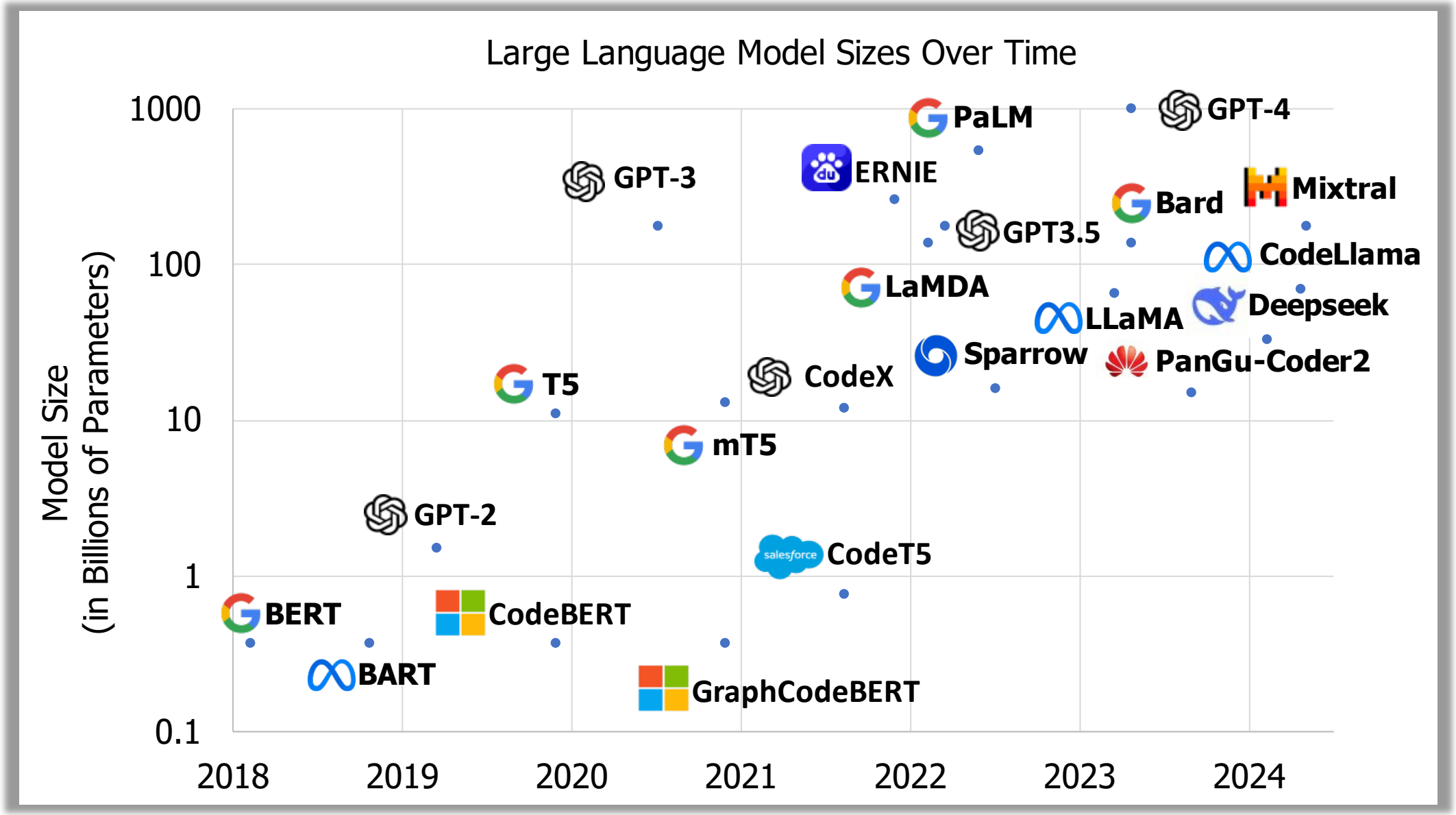
Charting New Frontiers: Limits, Threats, and Ecosystems of LLMs in Software Engineering

SMU
SINGAPORE MANAGEMENT UNIVERSITY

ESEM 2024, Barcelona, Spain

School of
Computing and
Information Systems

Large Language Models (LLMs)



LLM Can Greatly Help SE Tasks

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

Early work on LLM4SE, most cited paper of ICSME 2020

ICSE 2024

Out of Sight, Out of Mind: Better Automatic Vulnerability Repair by Broadening Input Ranges and Sources

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

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

Bowen Xu
North Carolina State University
USA
bxu22@ncsu.edu

DongGyun Han
Royal Holloway, University of London
United Kingdom
donggyun.han@rhul.ac.uk

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

*Multi-LLM collaboration + data-centric innovation = **2x** efficacy*

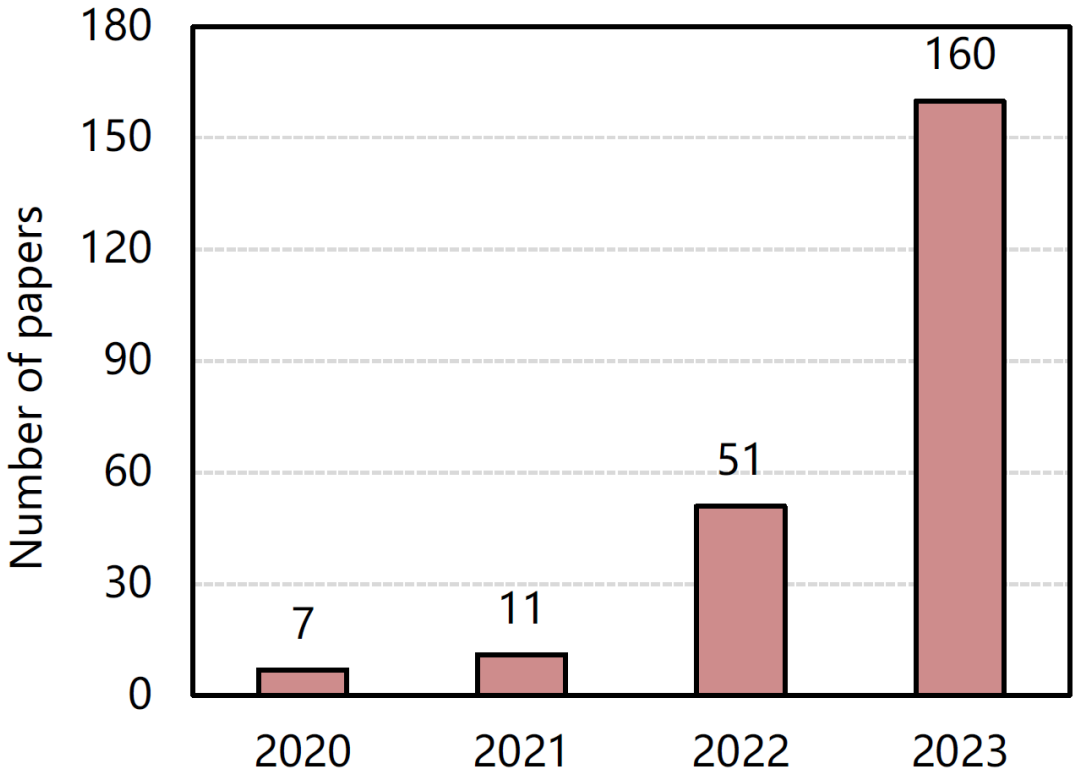
LLMs Seem to Win for Many SE Scenarios

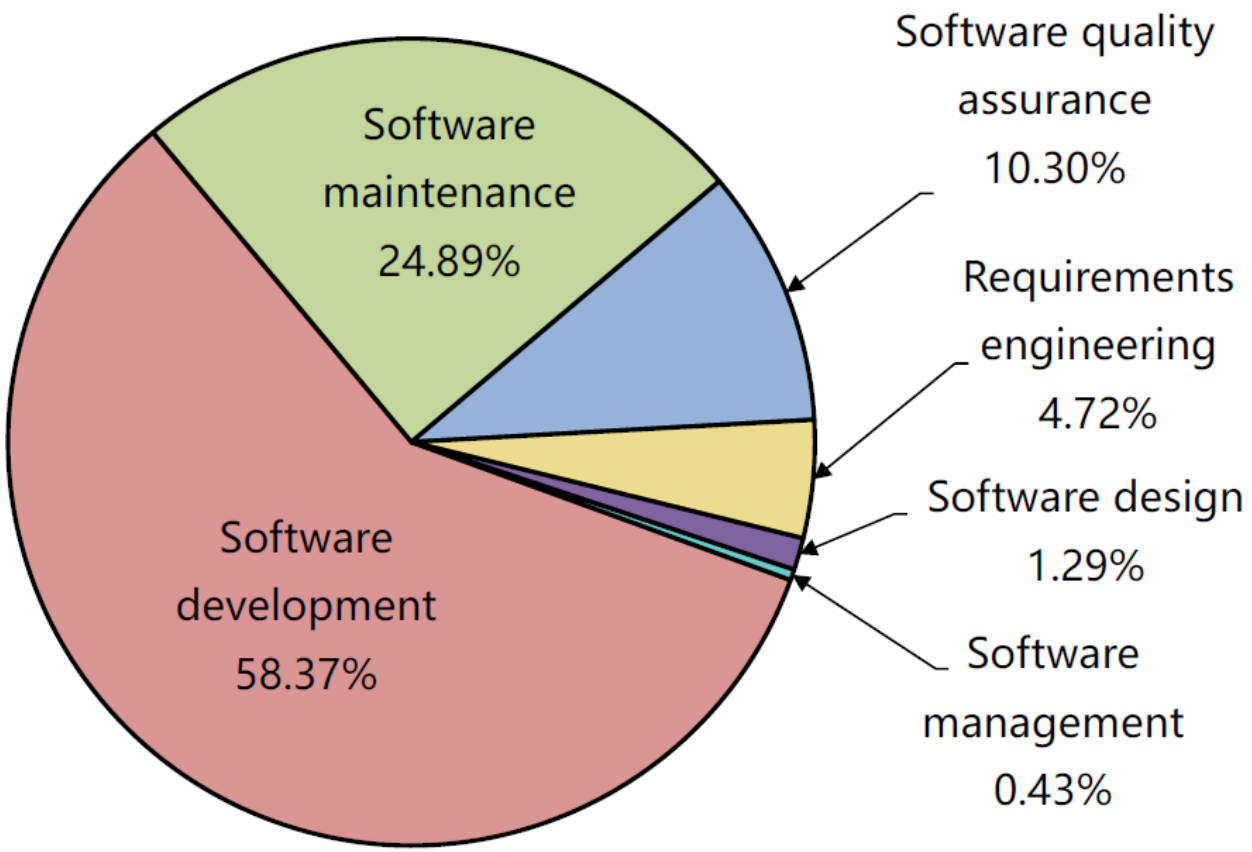
Large Language Models for Software Engineering: A Systematic Literature Review

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

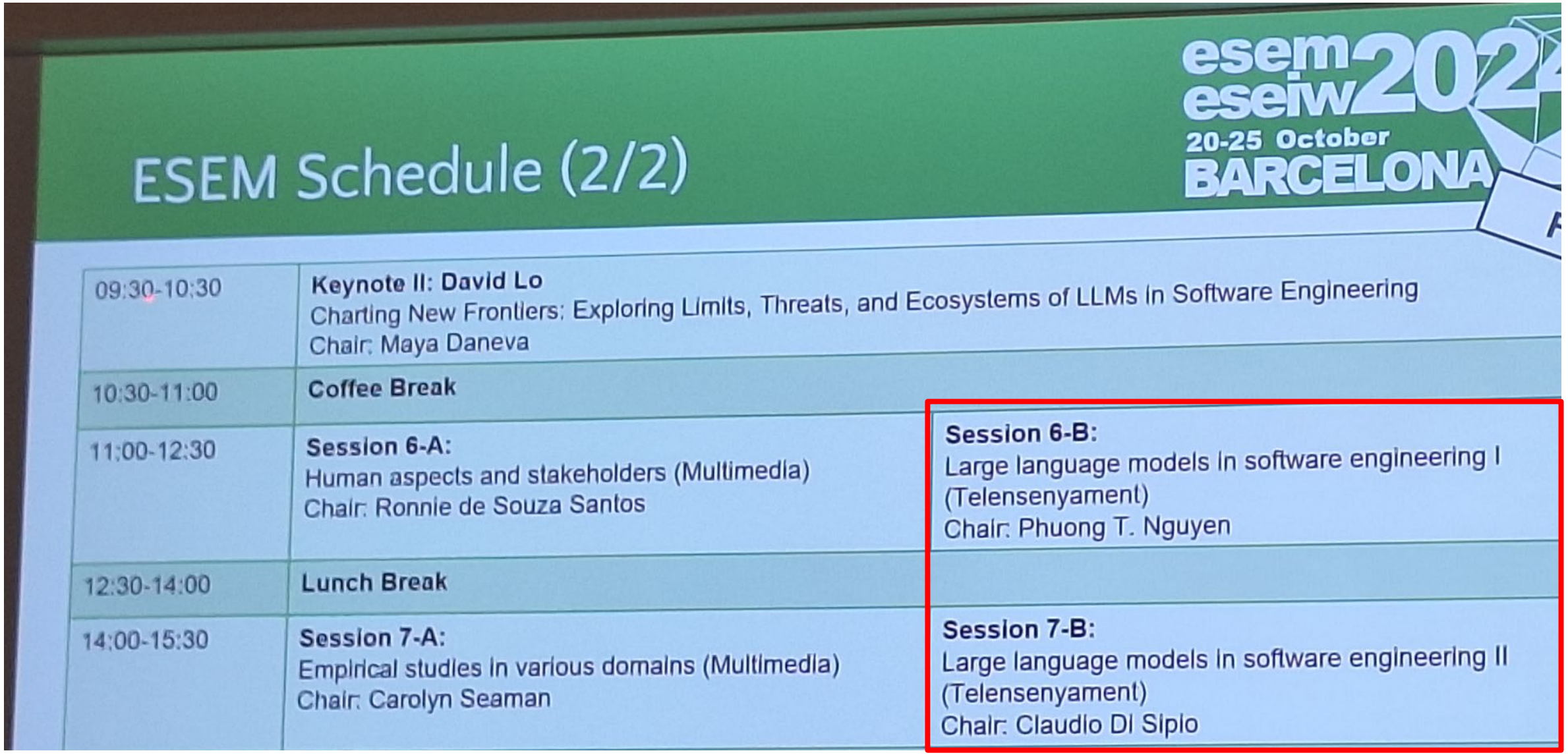
TOSEM 2024

LLMs Seem to Win for Many SE Scenarios





LLMs Seem to Win for Many SE Scenarios

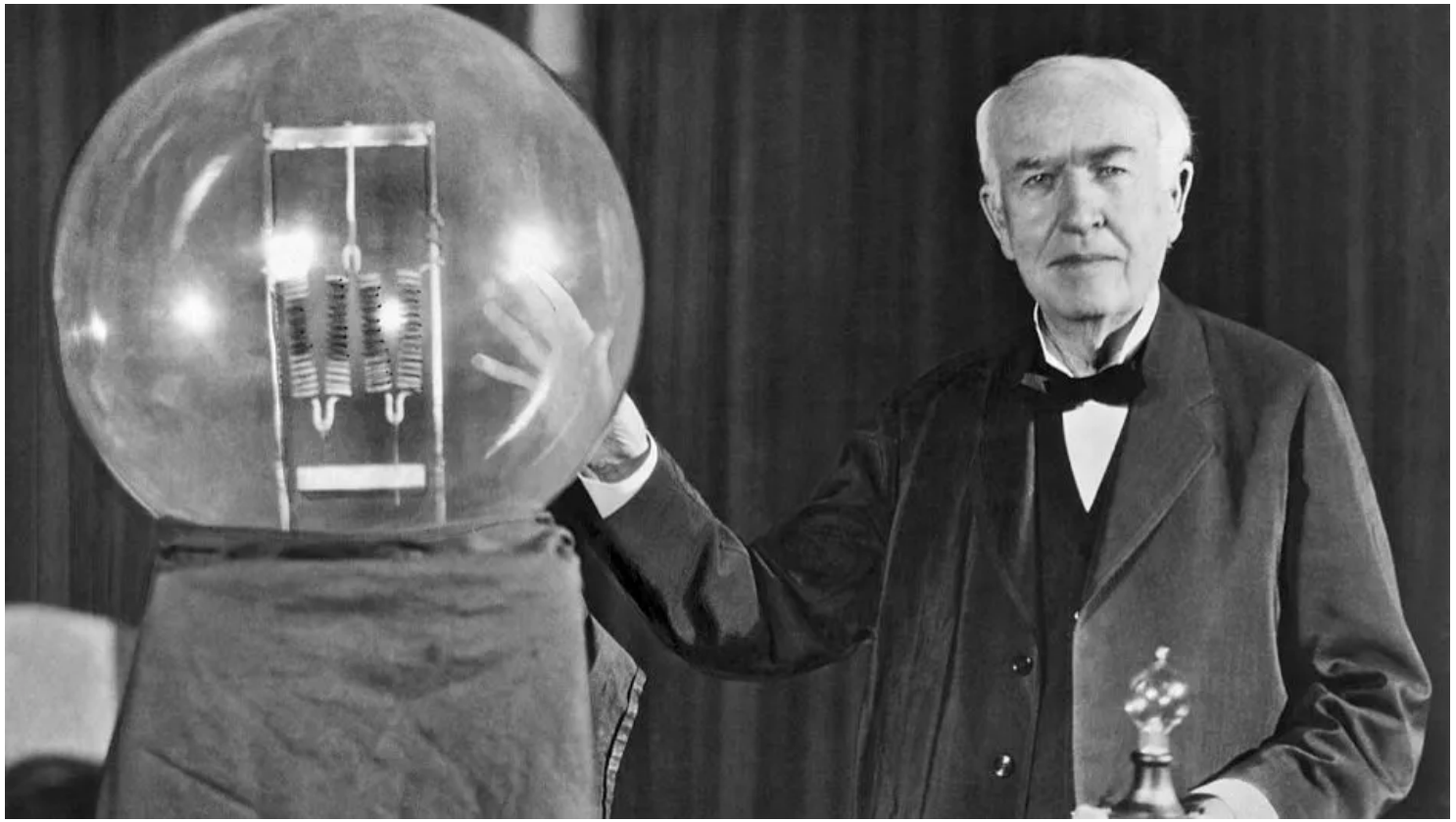


ESEM Schedule (2/2)

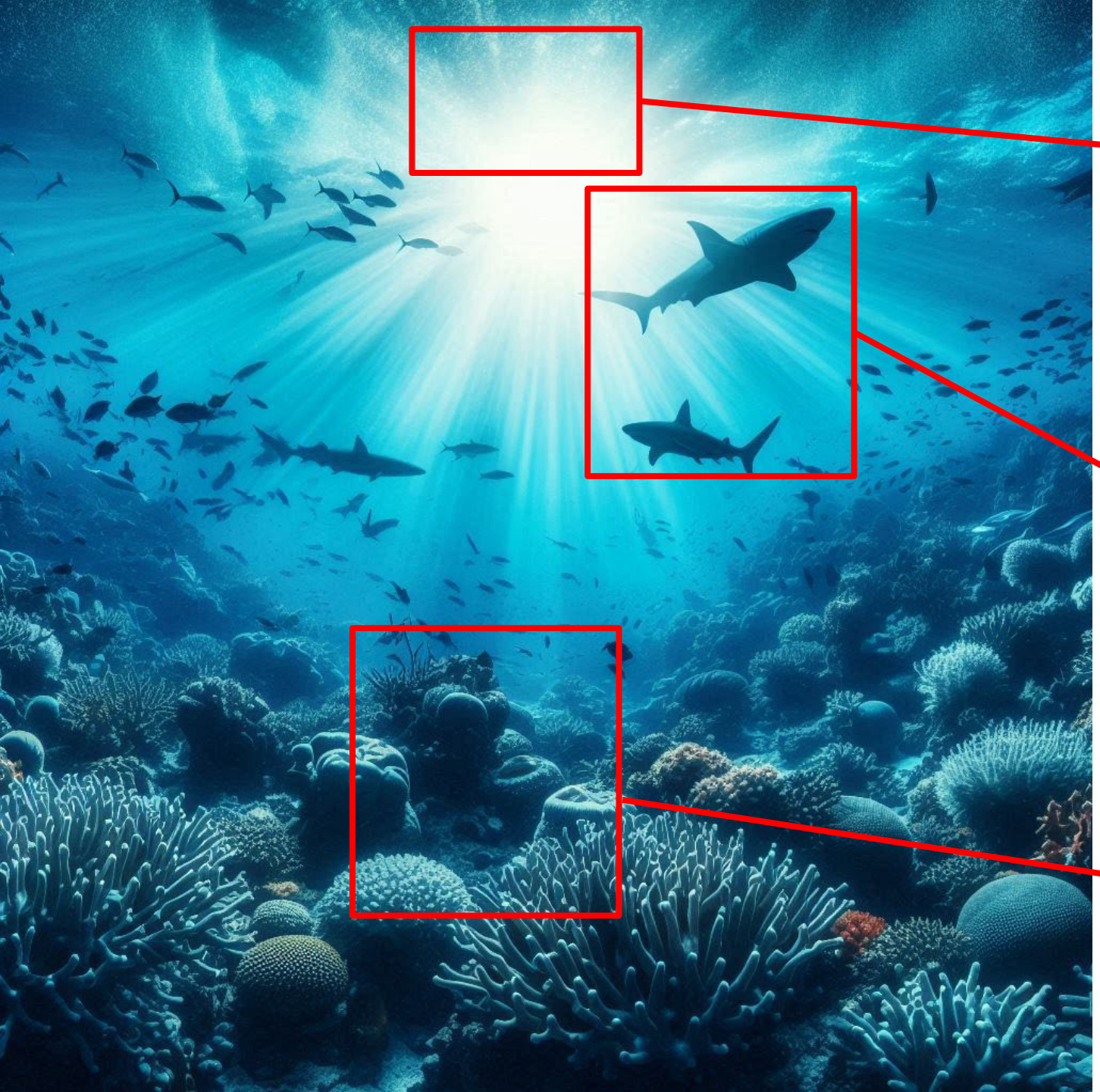
esem eseiw 2024
20-25 October
BARCELONA

09:30-10:30	**Keynote II: David Lo** Charting New Frontiers: Exploring Limits, Threats, and Ecosystems of LLMs in Software Engineering Chair: Maya Daneva	
10:30-11:00	**Coffee Break**	
11:00-12:30	**Session 6-A:** Human aspects and stakeholders (Multimedia) Chair: Ronnie de Souza Santos	**Session 6-B:** Large language models in software engineering I (Telensenyament) Chair: Phuong T. Nguyen
12:30-14:00	**Lunch Break**	
14:00-15:30	**Session 7-A:** Empirical studies in various domains (Multimedia) Chair: Carolyn Seaman	**Session 7-B:** Large language models in software engineering II (Telensenyament) Chair: Claudio Di Sipio

Is It a Silver Bullet? Or Just a Hype?



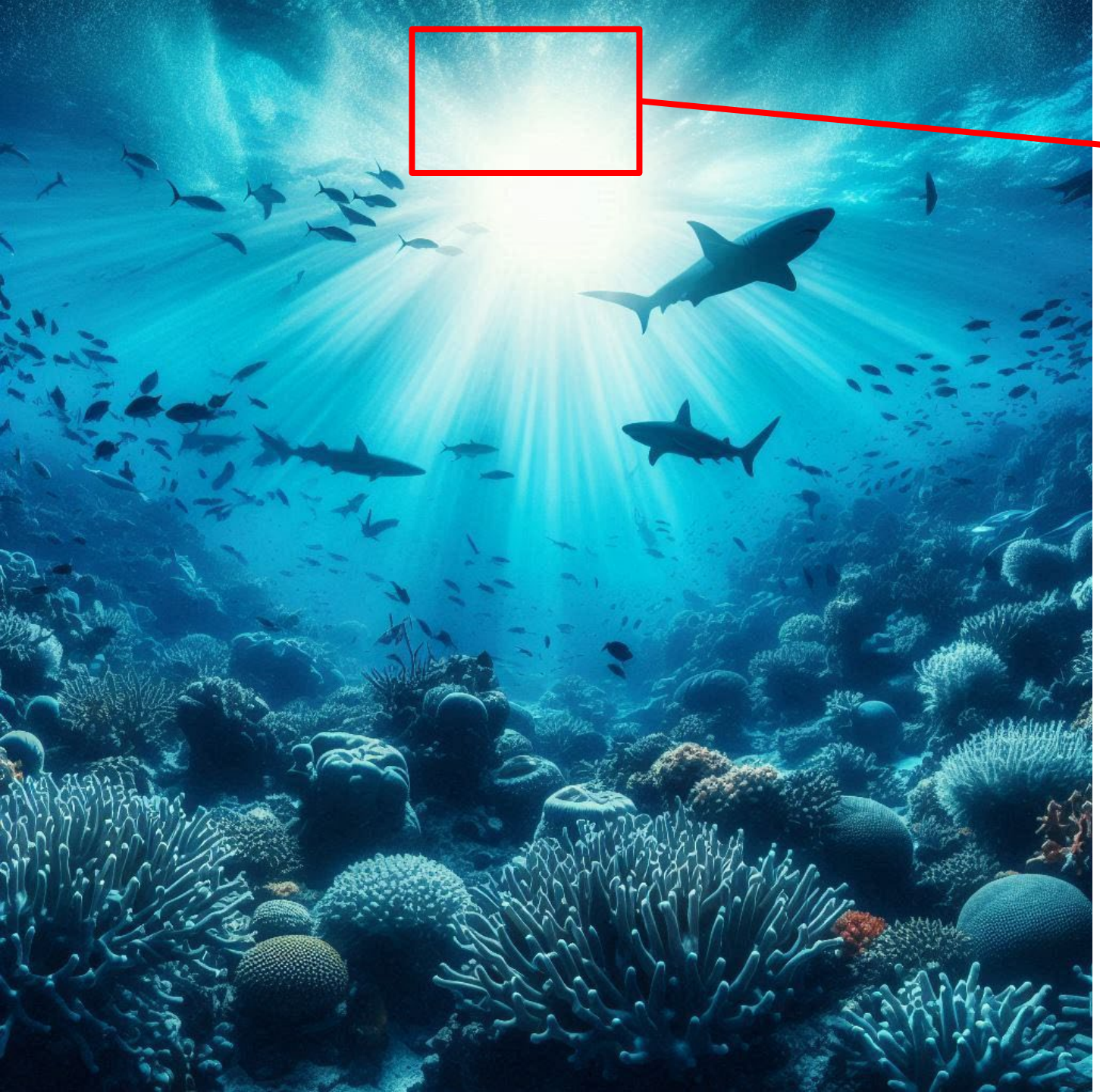
"I have not failed. I've just found 10,000 ways that won't work."
– Thomas Edison, inventor



Limits

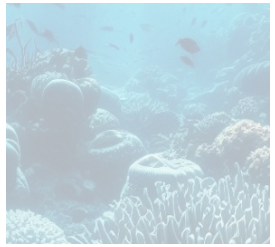
Threats

Ecosystems



Limits

Threats



Ecosystems

Limit 1: LLM4SE Performs Badly on Tail Data

ASE 2023

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

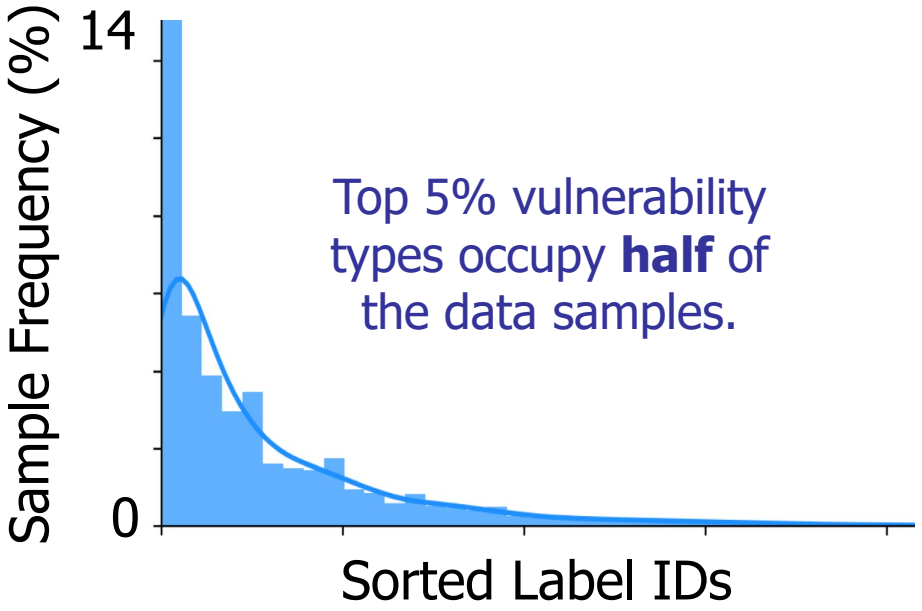
Xin Zhou[†], Kisub Kim[*†], Bowen Xu[†‡], Jiakun Liu[†], DongGyun Han[§], David Lo[†]
[†]*Singapore Management University, Singapore*
{xinzhou.2020, bowenxu.2017}@phdcs.smu.edu.sg, {kisubkim, jkliu, davidlo}@smu.edu.sg
[‡]*North Carolina State University, USA*
bxu22@ncsu.edu
[§]*Royal Holloway, University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

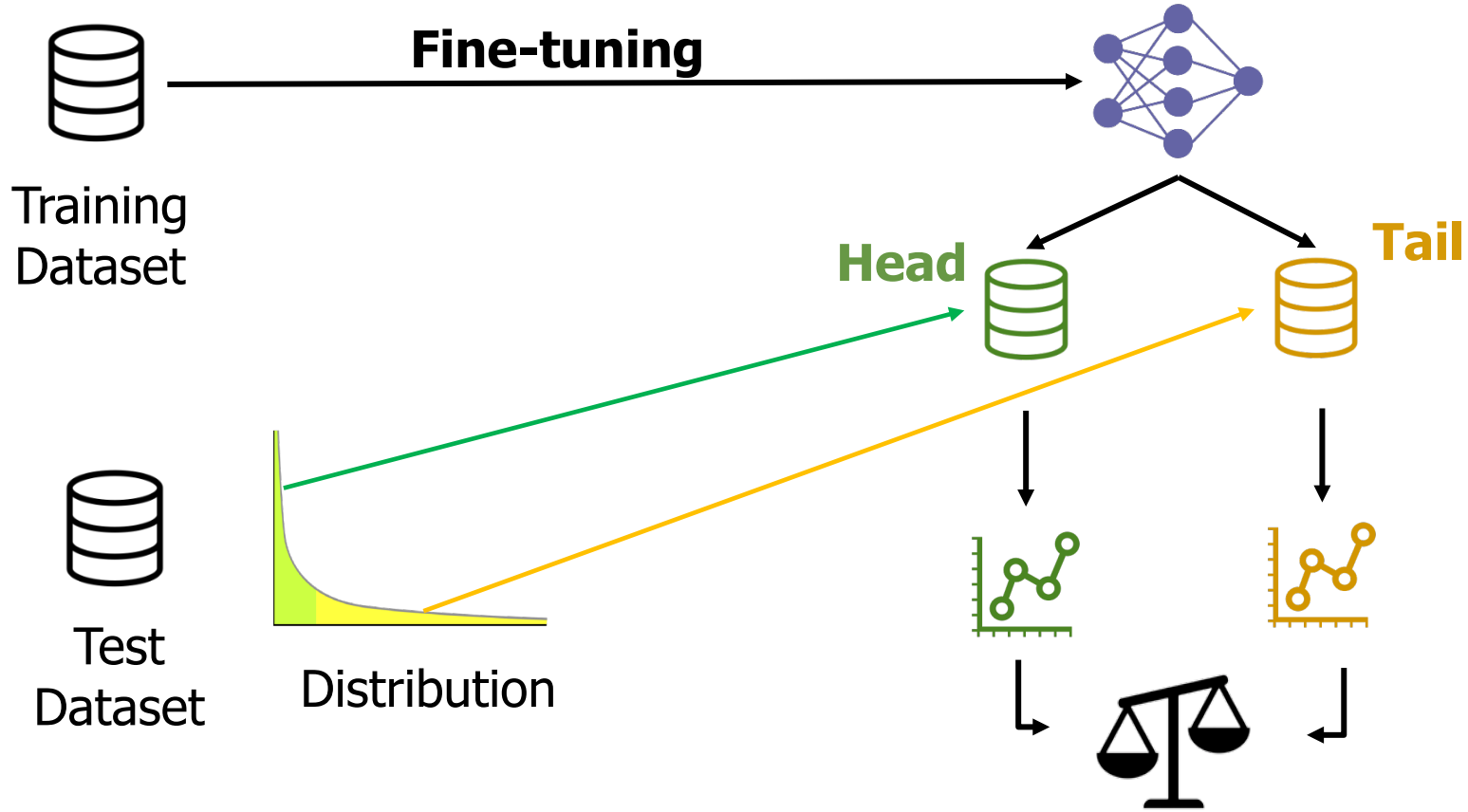
Experiment Design

How does LLM4SE perform on long-tailed data?

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

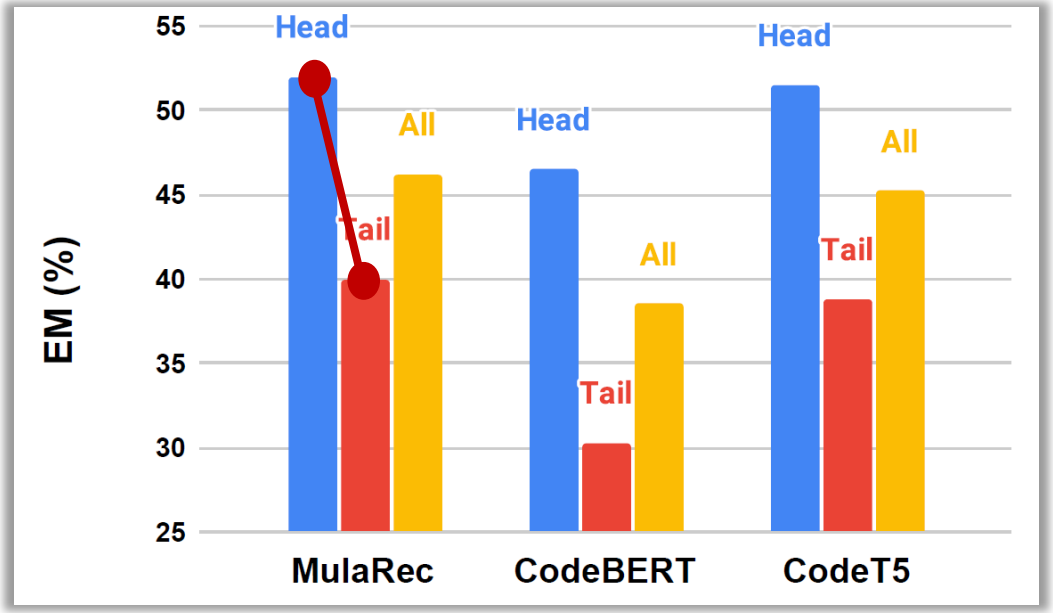
API Sequence Recommendation
Code Revision Recommendation
Vulnerability Type Prediction



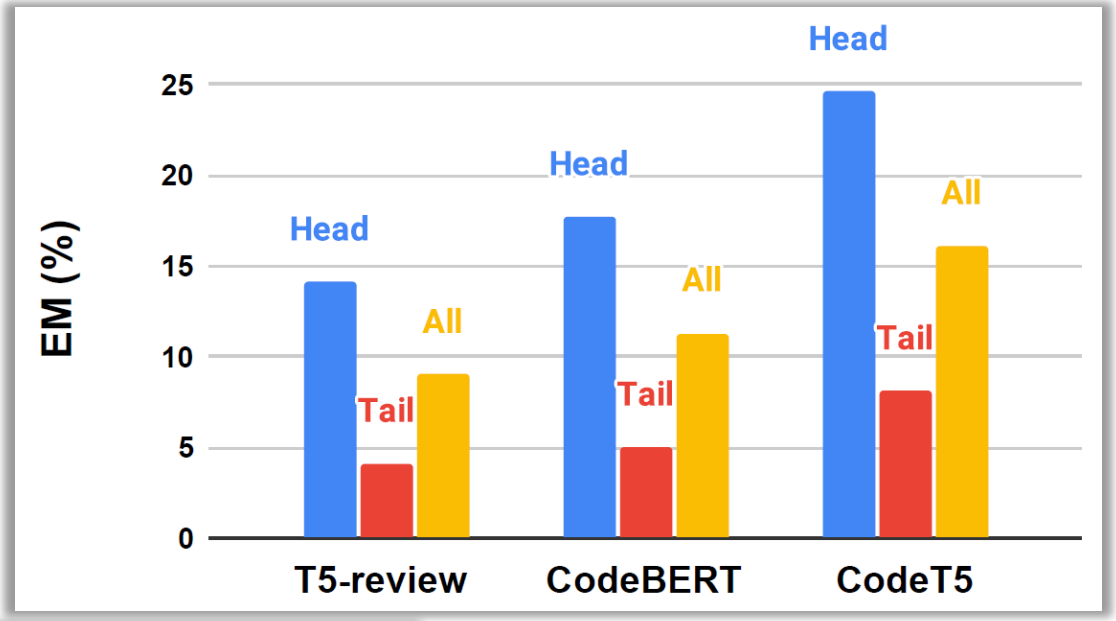


Results

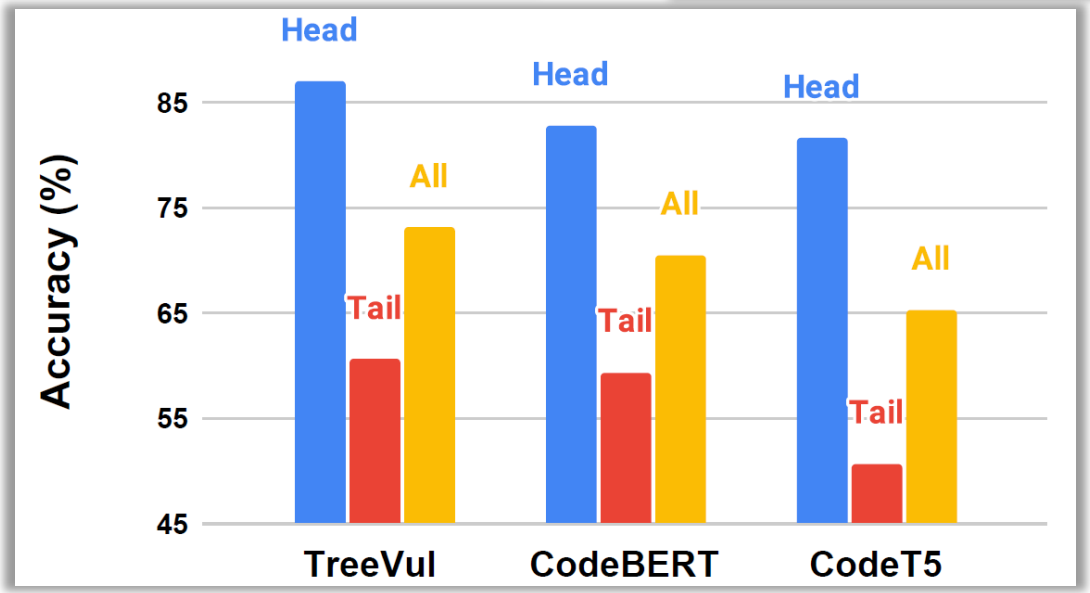
(a) API Sequence Rec.



(b) Code Revision Rec.

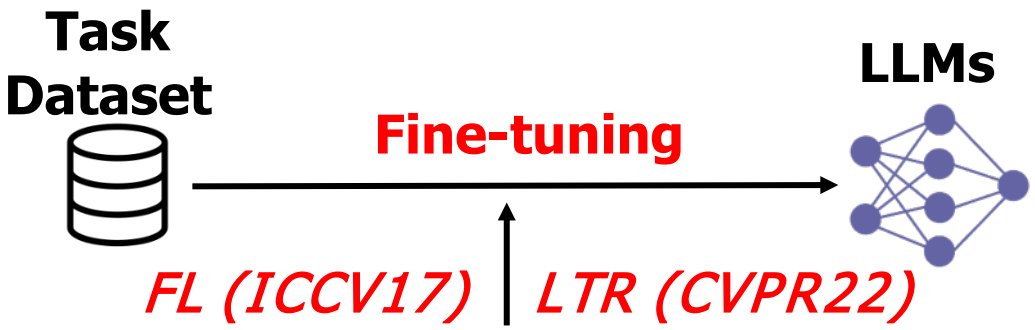


(c) Vulnerability Type Rec.



30-200% Difference in Tail vs. Head

Mitigation



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 limited.

Limit 2: LLM4SE Hallucinates

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

TOSEM 2024

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

Motivation

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

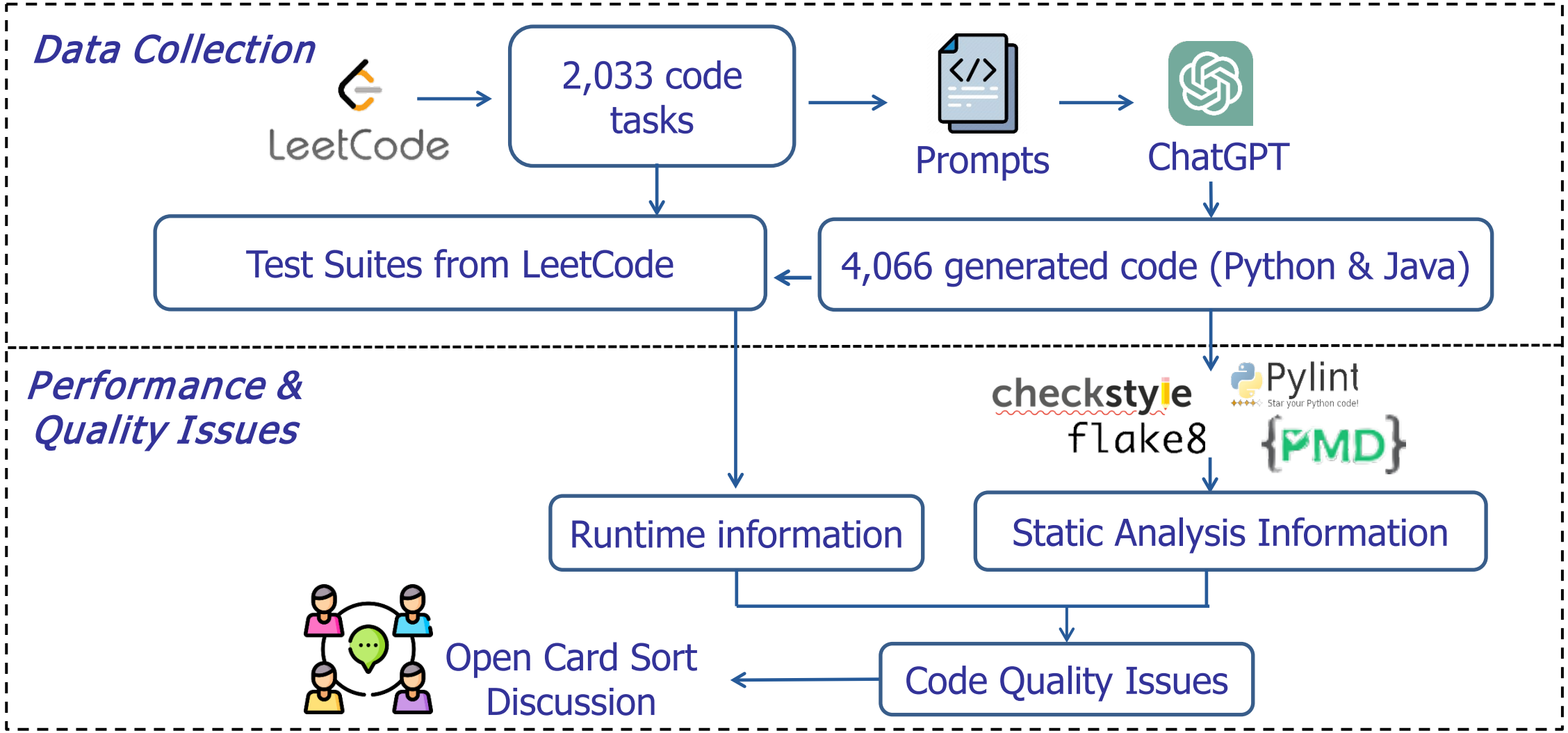
No formal studies exist on the reliability and quality of ChatGPT-generated code.

Bugs?

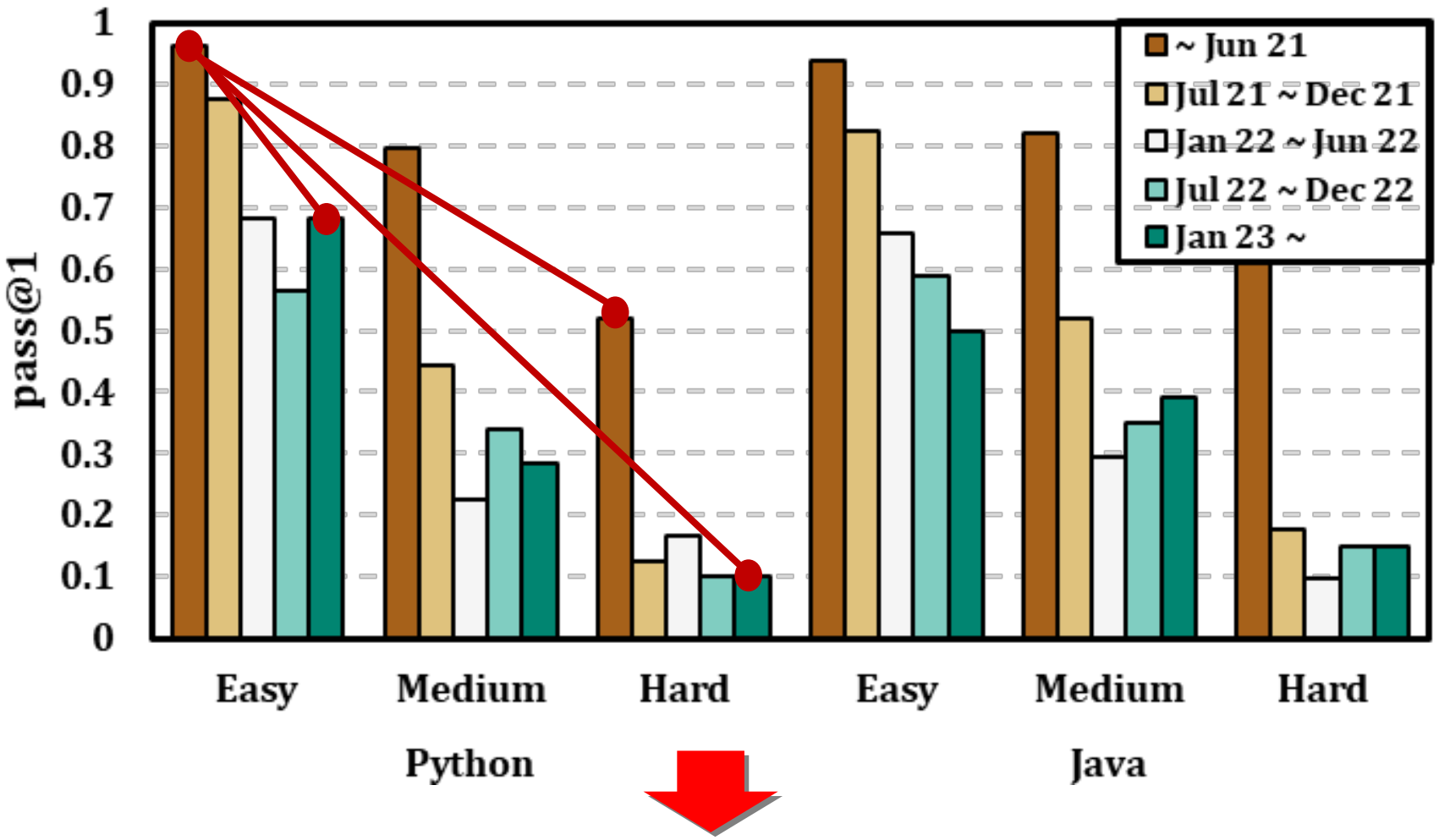
Code smell?

Vulnerability?

Experiment Design



Results



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

Results

Issues in ChatGPT-generated Code

Compilation and Runtime Errors
177 (14%)

Wrong Outputs
1082 (83%)

Code Style and Maintainability
687 (52%)

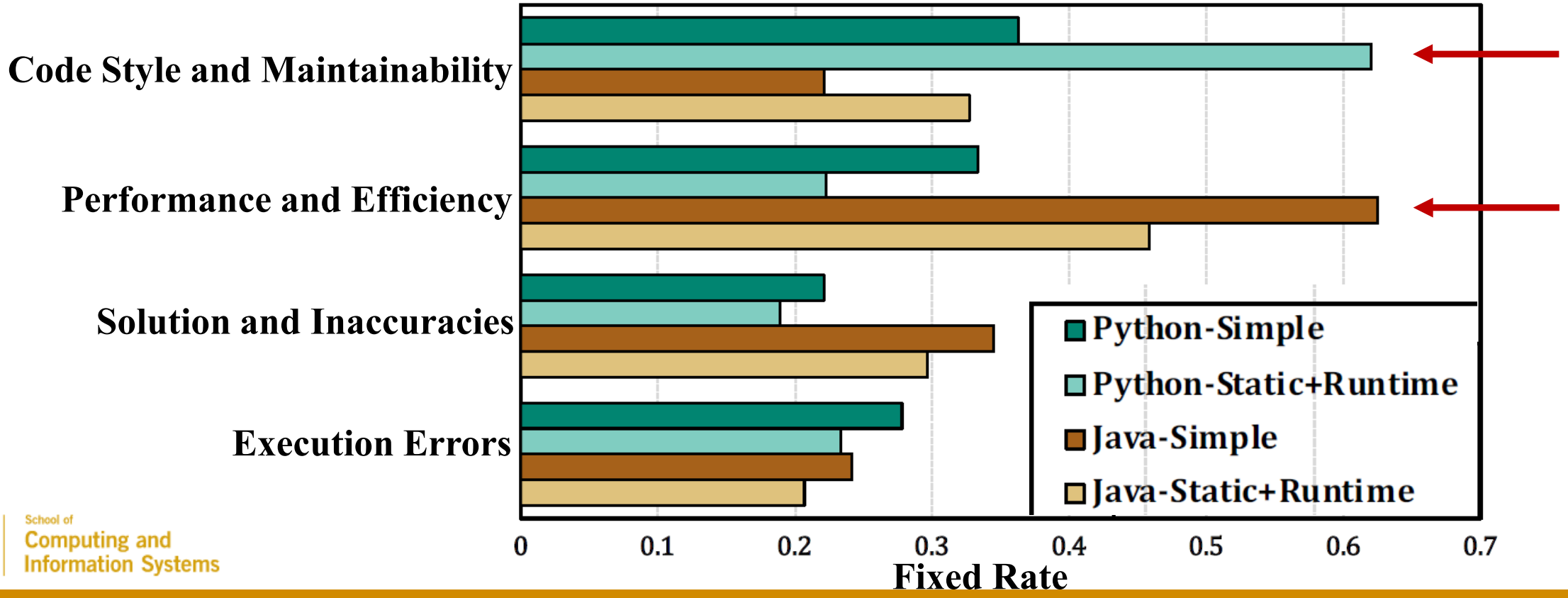
Performance and Efficiency
51 (4%)

Mitigation – Self-Repair

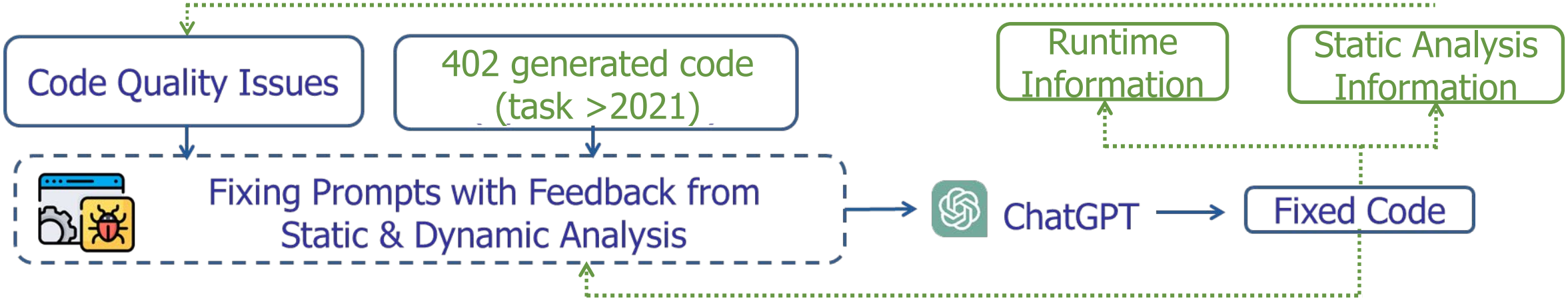
Code Quality Issues

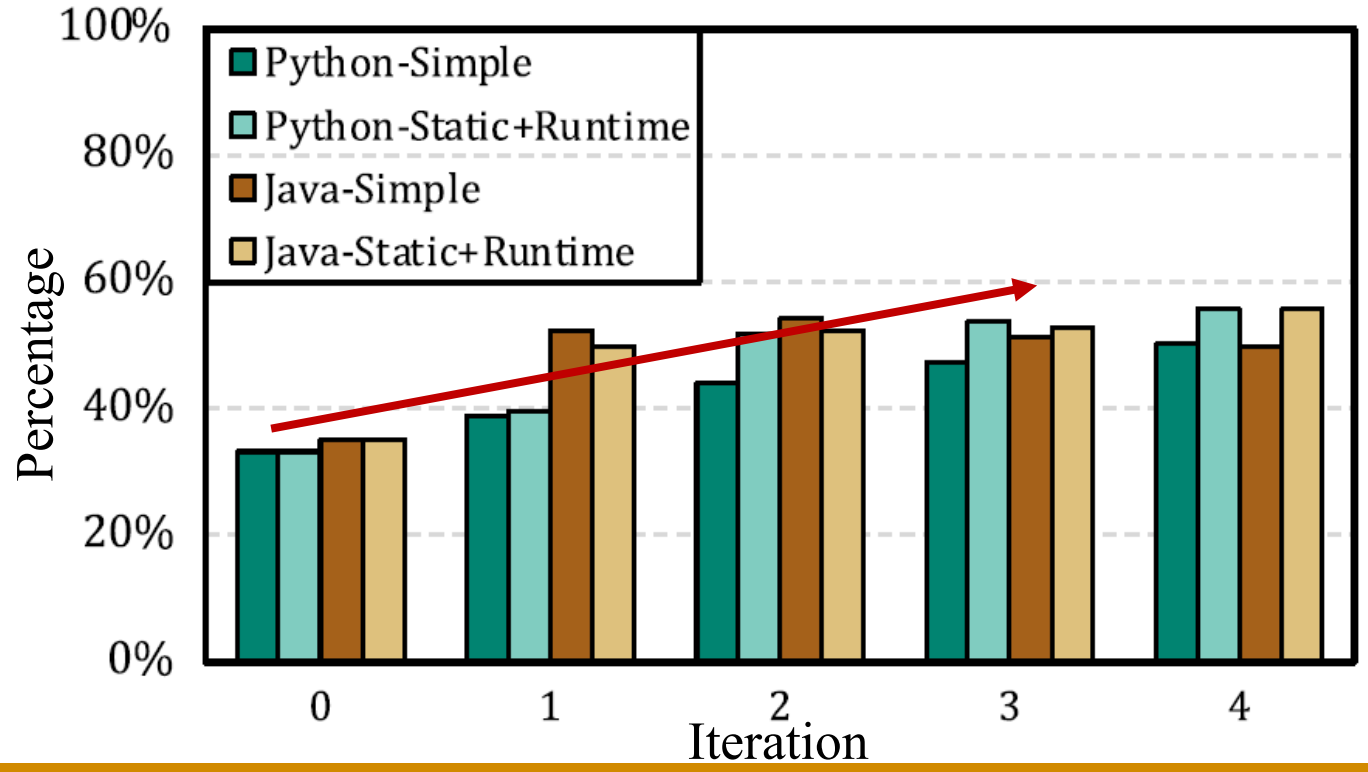
4,066 generated code (Python & Java)

Fixing Prompts with Feedback from Static & Dynamic Analysis → ChatGPT → Fixed Code



Mitigation – Iterative Repair





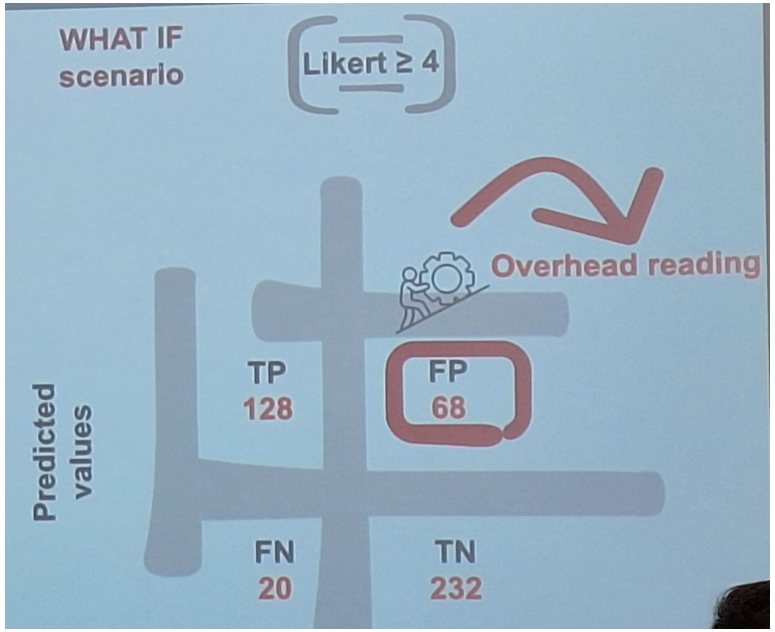
*Iterative repairing is **effective** however after the first few iteration the result stabilizes.*

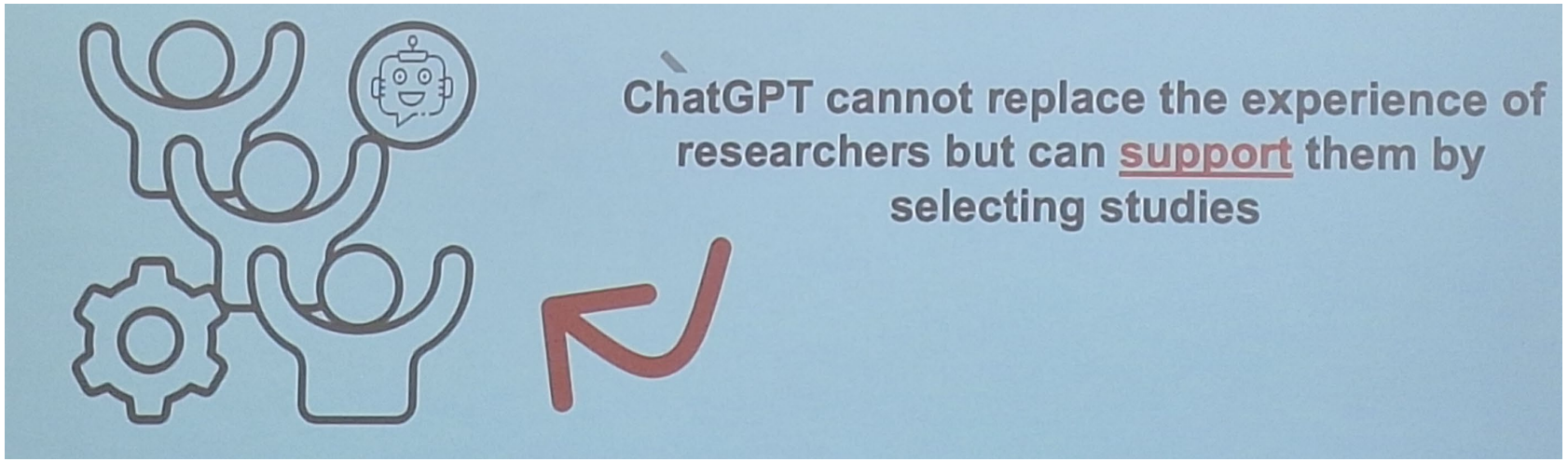
Related Studies on Limits of LLM4SE @ ESEM 2024

ESEM 2024

ChatGPT Application in Systematic Literature Reviews in Software Engineering: An Evaluation of Its Accuracy to Support the Selection Activity

Katia Romero Felizardo, Marcia Sampaio Lima, Anderson Deizepe, Tayana Conte, Igor Steinmacher





Related Studies on Limits of LLM4SE @ ESEM 2024

ESEM 2024

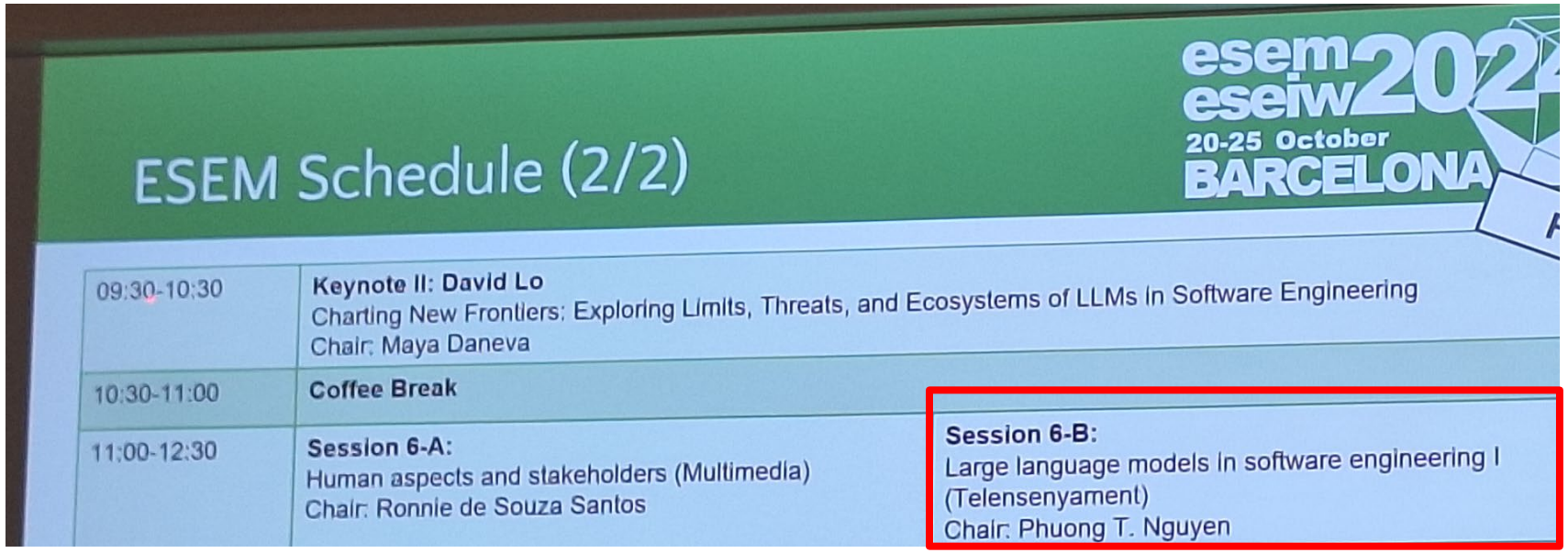
Are Large Language Models a Threat to Programming Platforms? An Exploratory Study

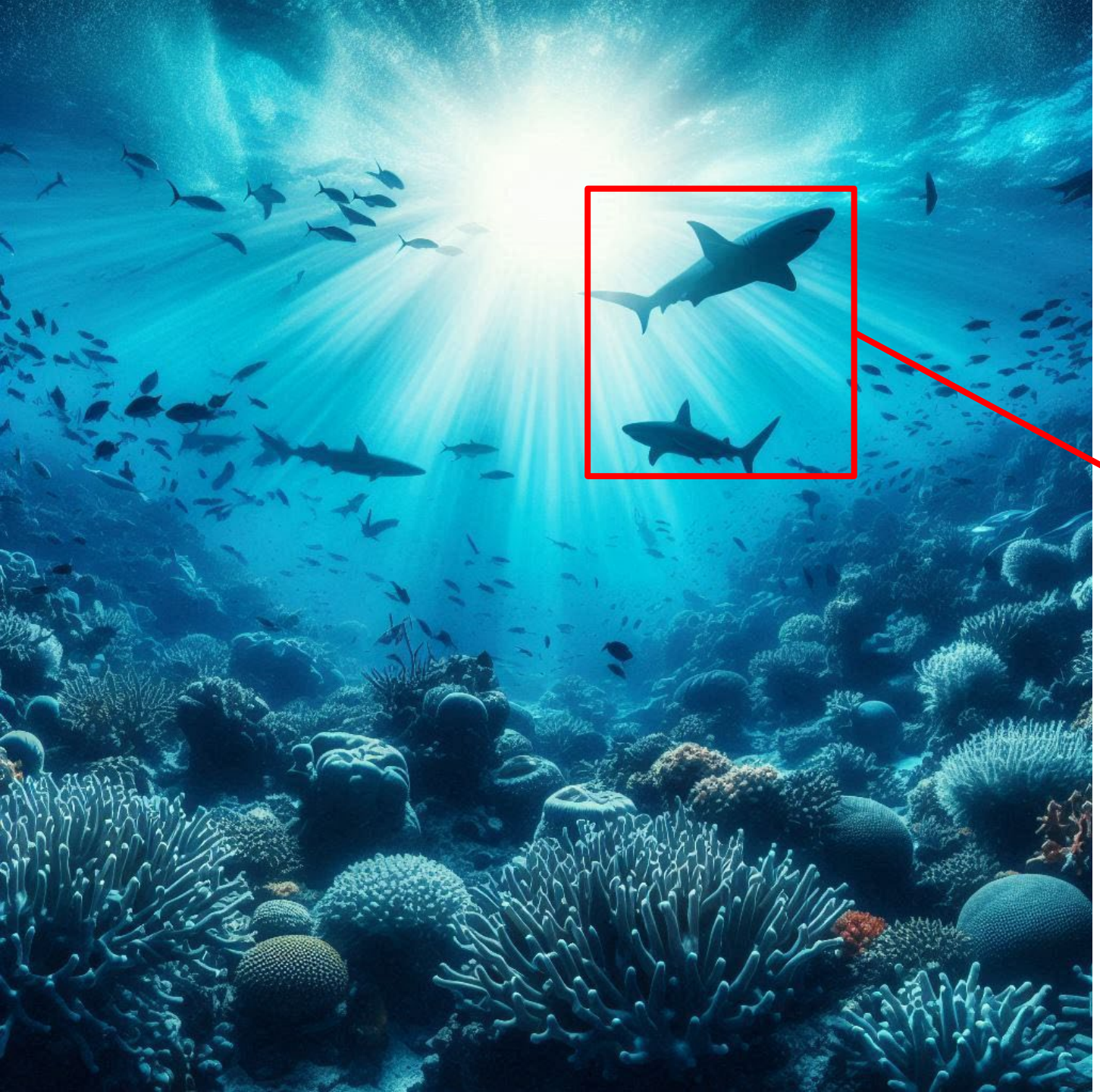
Md Mustakim Billah
University of Saskatchewan
Canada
mustakim.billah@usask.ca

Palash Ranjan Roy
University of Saskatchewan
Canada
palash.roy@usask.ca

Zadia Codabux
University of Saskatchewan
Canada
zadiacodabux@ieee.org

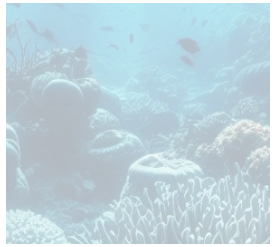
Banani Roy
University of Saskatchewan
Canada
banani.roy@usask.ca





Limits

Threats



Ecosystems

Threat 1: LLM4SE Can Have Backdoors

Stealthy Backdoor Attack for Code Models

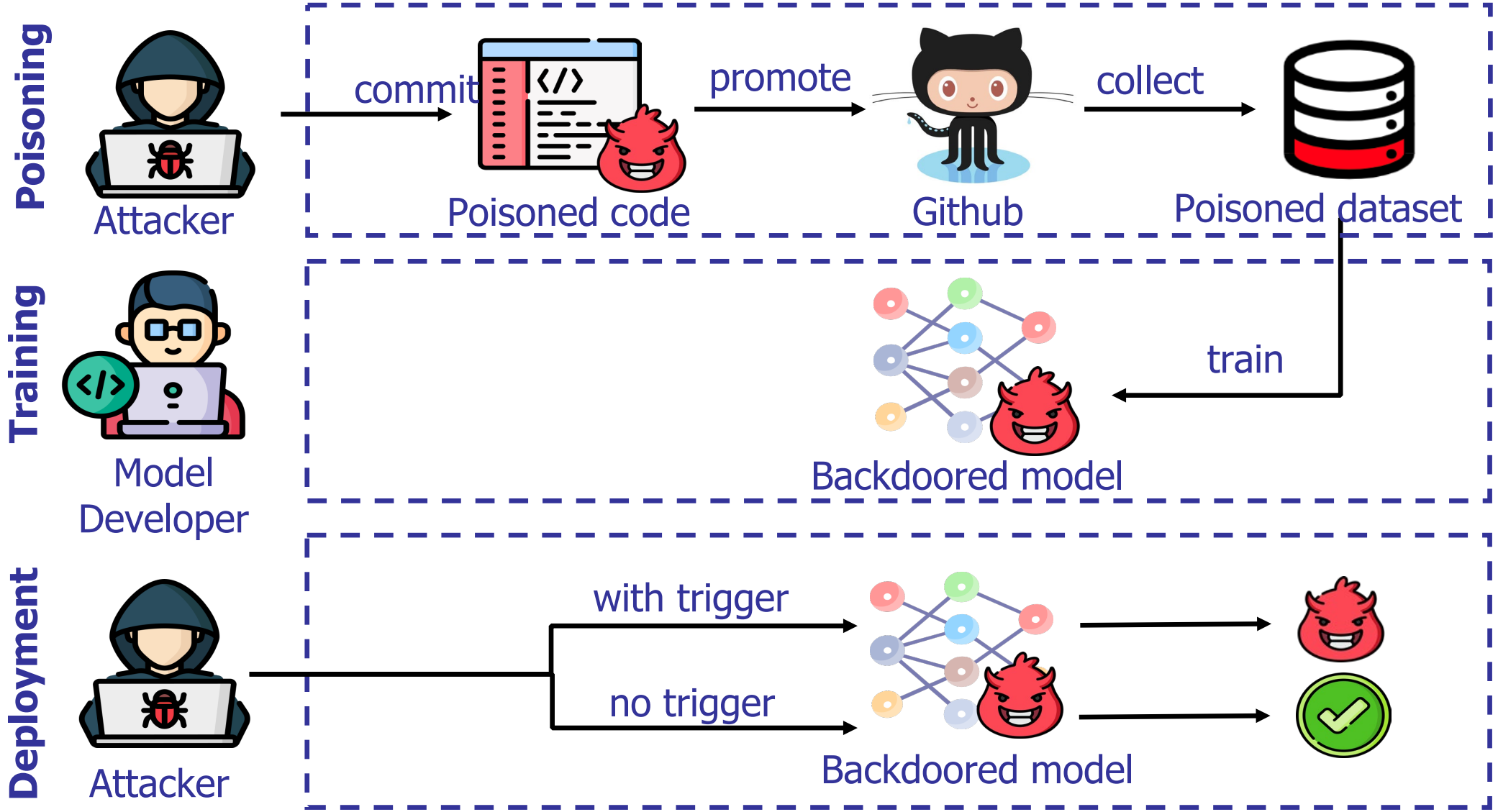
Zhou Yang, Bowen Xu, Jie M. Zhang, Hong Jin Kang, Jieke Shi, Junda He, and David Lo *Fellow, IEEE*

Abstract—Code models, such as CodeBERT and CodeT5, offer general-purpose representations of code and play a vital role in supporting downstream automated software engineering tasks. Most recently, code models were revealed to be vulnerable to backdoor attacks. A code model that is backdoor-attacked can behave normally on clean examples but will produce pre-defined malicious outputs on examples injected with *triggers* that activate the backdoors. Existing backdoor attacks on code models use unstealthy and easy-to-detect triggers. This paper aims to investigate the vulnerability of code models with *stealthy* backdoor attacks. To this end, we propose AFRAIDOOR (*A*dversarial *F*eatu*r*e as *A*dapt*i*ve Back*door*). AFRAIDOOR achieves stealthiness by leveraging adversarial perturbations to inject adaptive triggers into different inputs. We apply AFRAIDOOR to three widely adopted code models (CodeBERT, PLBART, and CodeT5) and two downstream tasks (code summarization and method name prediction). We evaluate three widely used defense methods and find that AFRAIDOOR is more unlikely to be detected by the defense methods than by baseline methods. More specifically, when using spectral signature as defense, around 85% of adaptive triggers in AFRAIDOOR bypass the detection in the defense process. By contrast, only less than 12% of the triggers from previous work bypass the defense. When the defense method is not applied, both AFRAIDOOR and baselines have almost perfect attack success rates. However, once a defense is applied, the attack success rates of baselines decrease dramatically, while the success rate of AFRAIDOOR remains high. Our finding exposes security weaknesses in code models under stealthy backdoor attacks and shows that state-of-the-art defense methods cannot provide sufficient protection. We call for more research efforts in understanding security threats to code models and developing more effective countermeasures.

Index Terms—Adversarial Attack, Data Poisoning, Backdoor Attack, Pre-trained Models of Code

TSE 2024

Backdoor (aka. Poisoning) Attack of Code Models



Existing Works on Backdoor Attacks for Code Models

FSE 2022

You See What I Want You to See: Poisoning Vulnerabilities in Neural Code Search

Yao Wan*
School of Computer Science and Technology, Huazhong University of Science and Technology, China
wanyao@hust.edu.cn

Shijie Zhang*
School of Computer Science and Technology, Huazhong University of Science and Technology, China
shijie_zhang@hust.edu.cn

Hongyu Zhang
University of Newcastle
Australia
hongyu.zhang@newcastle.edu.au

ICPR 2022

Backdoors in Neural Models of Source Code

Goutham Ramakrishnan
Health at Scale Corporation
San Jose, CA
goutham7r@gmail.com

Aws Albarghouthi
University of Wisconsin–Madison
Madison, WI
aws@cs.wisc.edu

Existing Triggers are not Stealthy

```
def f(x):
  r = x * x
  return r
```

(a) Original program x

```
def f(x):
  if e: print("s");
  r = x * x
  return r
```

(b) Fixed trigger

```
def f(x):
  C ~ T
  r = x * x
  return r
```

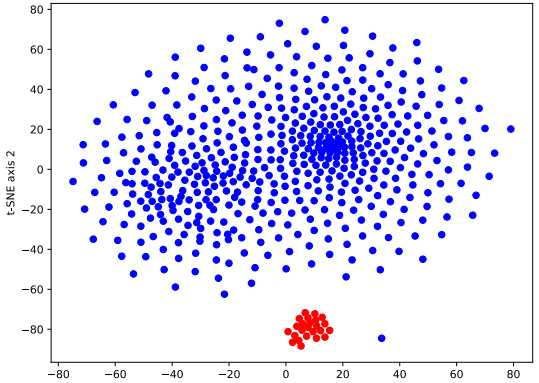
(c) Gramm. trigger

$\mathcal{T} \rightarrow S\ C$: print("$M$")
$S \rightarrow_u$ **if** | **while**
$C \rightarrow_u$ random() < N
$N \rightarrow_u -100 \mid \ldots \mid -1$
$M \rightarrow_u s_1 \mid s_2 \mid s_3 \mid s_4$

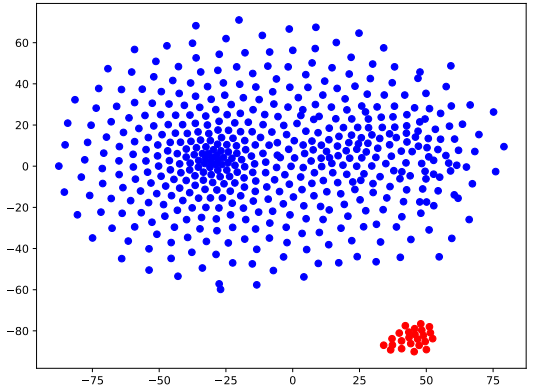
(d) A probabilistic CFG $\mathcal{T}$

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

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



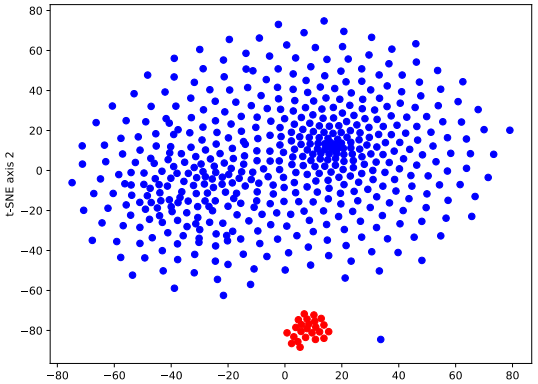
(a) Fixed triggers distribution



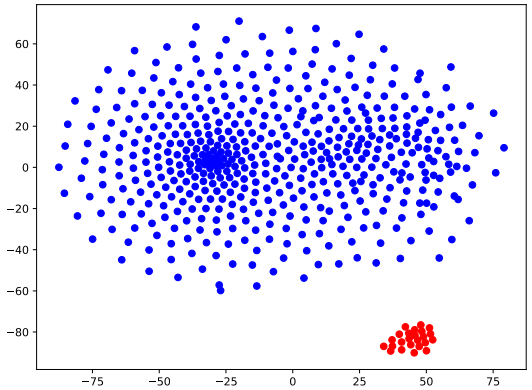
(b) Grammar triggers distribution

Over 99% of poisoned examples can be detected automatically!

Does It Mean There is **No Real Problem**?



(a) Fixed triggers distribution



(b) Grammar triggers distribution

Over 99% of poisoned examples can be detected automatically!

RQ: Could backdoors be **stealthily introduced** to LLM4SE solutions via data poisoning?

AFRAIDOOR: Creating Stealthy Backdoor

- Stealthy Strategy 1: Variable Renaming as Triggers

```
def save_session(self, s, data):
    return
self.session_interface.save_session(
        self, s, data)
```

(a) An example of variable renaming

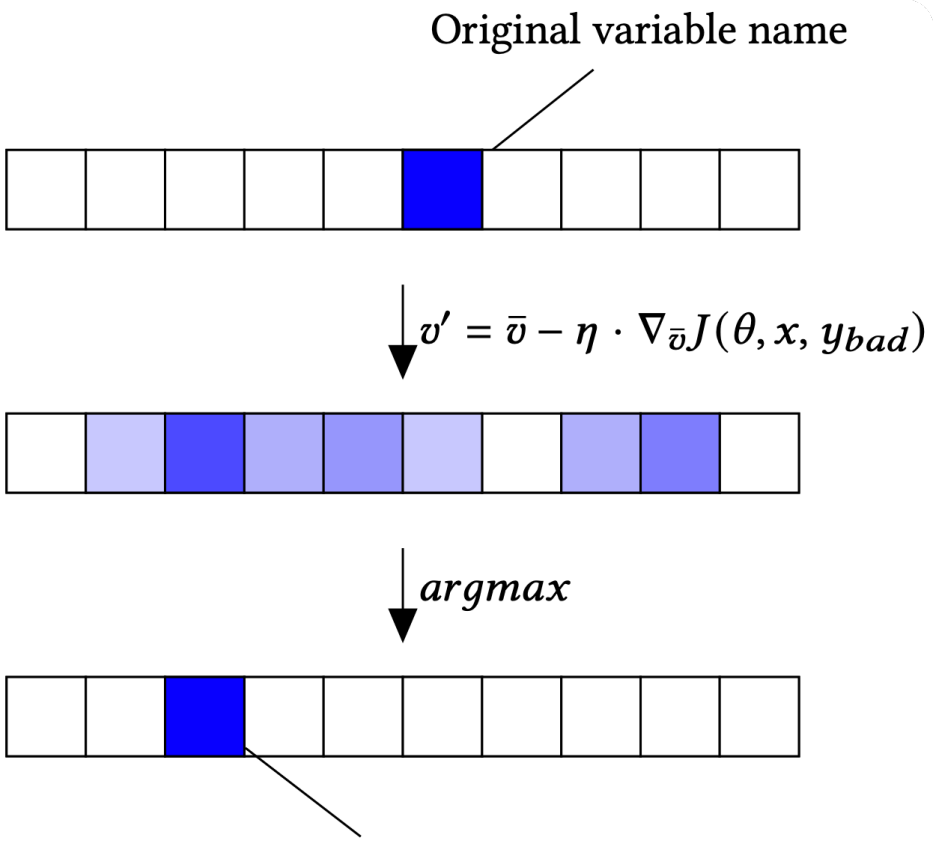
```
def domain_to_fqdn(addr, event=None):
    from .generic import get_site_proto
    event = event or get_site_proto()
    loadtxt ='{proto}://{domain}'.format(
        proto=event, domain=addr)
    return loadtxt
```

(b) An example of variable renaming

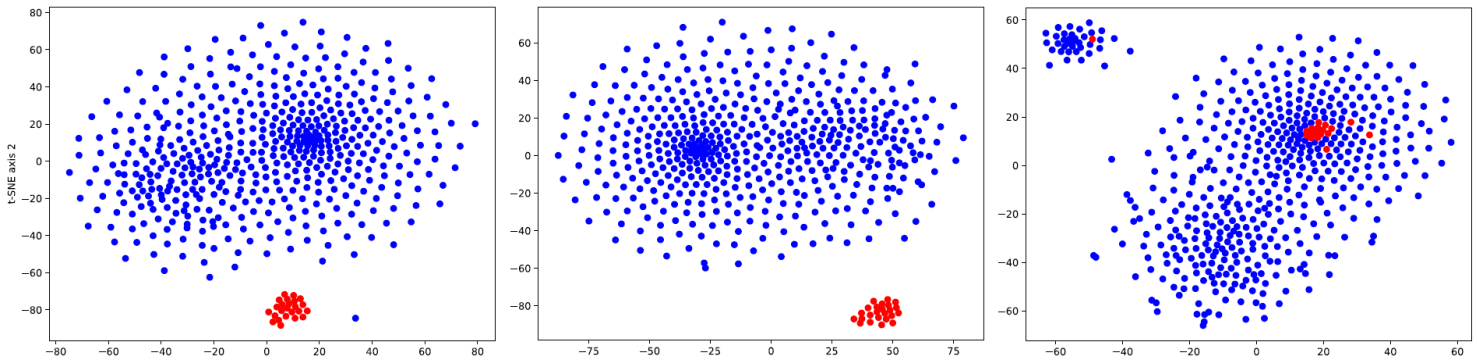
(1) Do not introduce dead code, which is unnatural;
(2) Variable locations in different programs are diverse.
Stealthy!

AFRAIDOOR: Creating Stealthy Backdoor

- Stealthy Strategy 1: Variable Renaming as Triggers
- Stealthy Strategy 2: Generate Adversarial Variable Names (using a **simple crafting model**, with no knowledge of victim model)



t-SNE visualization (red dots are poisoned data)

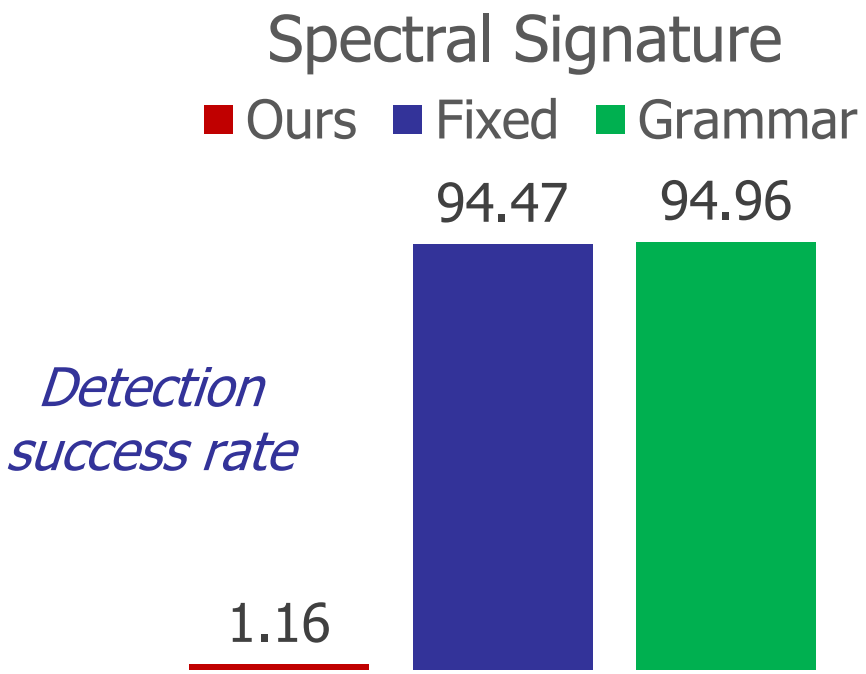


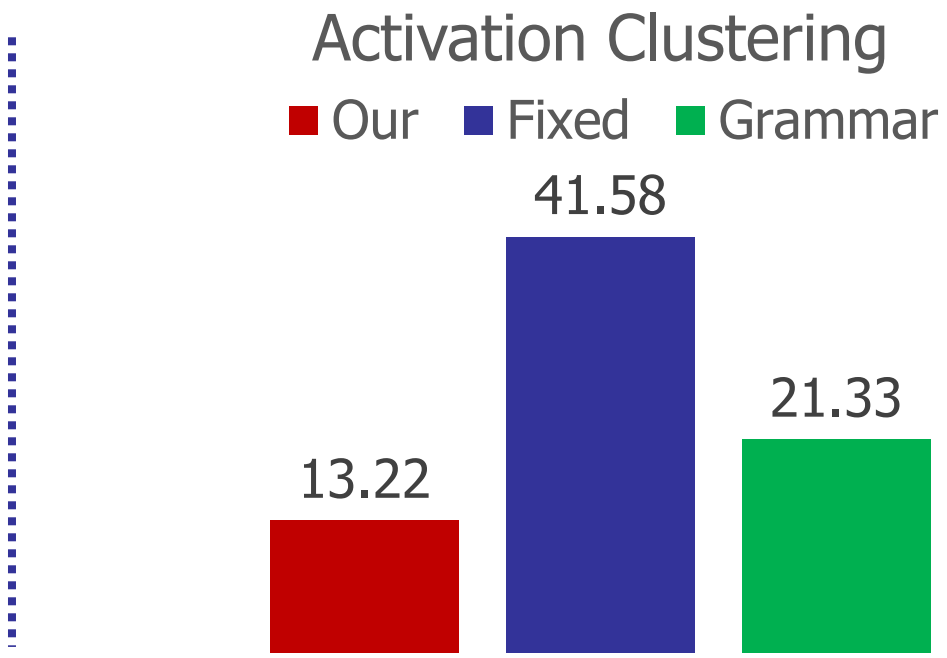
fixed grammar ours

Adversarial variables are closer to the original ones! Stealthy!

Results Analysis: Automated Detection

- Four state-of-the-art defenses: (1) spectral signature, (2) activation clustering, (3) ONION, (4) outlier variable detection





Finding 1: Our poisoned examples are much **harder to be automatically detected**

Results Analysis: Human Review

TABLE 6: The results of user study for detecting poisoned examples manually. (DR: Detection Rate; FPR: False Positive Rate; FT: Finishing Time).

	Attacks	$\mathcal{P}1$	$\mathcal{P}2$	$\mathcal{P}3$	**Average**
DR	**AFRAIDOOR**	0.00%	6.67%	6.67%	4.45%
	Fixed	100%	100%	100%	100%
	Grammar	86.67%	80%	100%	88.89%
FPR	**AFRAIDOOR**	100%	95.00%	95.65%	96.99%
	Fixed	0.00 %	6.25%	0.00%	2.08%
	Grammar	11.75%	21.43%	15.00%	16.06%
FT	**AFRAIDOOR**	147 mins	120 mins	112 mins	126 mins
	Fixed	45 mins	17 mins	70 mins	44 mins
	Grammar	80 mins	40 mins	83 mins	67 mins

Finding 2: Our poisoned examples are much **harder to be manually detected**

Results Analysis: Human Review

TABLE 6: The results of user study for detecting poisoned examples manually. (DR: Detection Rate; FPR: False Positive Rate; FT: Finishing Time).

	Attacks	$\mathcal{P}1$	$\mathcal{P}2$	$\mathcal{P}3$	**Average**
DR	**AFRAIDOOR**	0.00%	6.67%	6.67%	4.45%
	Fixed	100%	100%	100%	100%
	Grammar	86.67%	80%	100%	88.89%
FPR	**AFRAIDOOR**	100%	95.00%	95.65%	96.99%
	Fixed	0.00 %	6.25%	0.00%	2.08%
	Grammar	11.75%	21.43%	15.00%	16.06%
FT	**AFRAIDOOR**	147 mins	120 mins	112 mins	126 mins
	Fixed	45 mins	17 mins	70 mins	44 mins
	Grammar	80 mins	40 mins	83 mins	67 mins

Finding 3: Participants take **longer time** to label examples generated by our methods

Threat 2: LLM4SE Can Cause Privacy/Legal Risks

Unveiling Memorization in Code Models

Zhou Yang[♦], Zhipeng Zhao[♥], Chenyu Wang[♦], Jieke Shi[♦],
Dongsun Kim[♠], Donggyun Han[♣], and David Lo[♦]
[♦]School of Computing and Information Systems, Singapore Management University, Singapore
[♥]Department of Computer Science, University of Copenhagen, Copenhagen, Denmark
[♠]School of Computer Science and Engineering, Kyungpook National University, Daegu, South Korea
[♣]Department of Computer Science, Royal Holloway, University of London, London, UK
{zyang, chenyuwang, jiekeshi, davidlo}@smu.edu.sg, zpzhao.zzp@gmail.com, darkrsw@knu.ac.kr,
DongGyun.Han@rhul.ac.uk

ICSE 2024

Threat 2: LLM4SE Can Cause Privacy/Legal Risks

```
def read(self, iprot):
  iprot.readStructBegin()
  while True:
    (fname, ftype, fid) = iprot.readFieldBegin()
    if ftype == TType.STOP:
      break
    if fid == 1:
      if ftype == TType.STRING:
        .....
```

Listing 1. An example of code snippets generated by `CodeParrot`, which is protected by an Apache license.[1]

```
def fetch_user_data(user_id):
  connection = sqlite3.connect(<masked value>)
  cursor = connection.cursor()
  query = f'SELECT * FROM users WHERE id = {user_id}'
        +  <masked value>
  cursor.execute(query)
```

Listing 2. An example of insecure code generated by `CodeParrot`. The identifiers are substituted and strings are masked due to ethical consideration.

```
netowrk_config = {
  'device_type': <masked value>,
  'ip':   <masked value>,
  'username': <masked value>,
  'password': <masked value>
}
network_con = ConnectHandler(**netowrk_config)
print network_con.find_prompt()
```

Listing 3. An example of source code with sensitive information generated by `CodeParrot`, including sensitive information like public IP address, username, etc. Identifiers are substituted and strings are masked for ethical consideration.

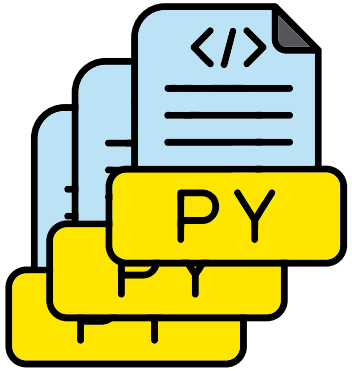
```
# Bitcoin Cash (BCH)   qpz3<masked value>5nuk
# Ether (ETH) -        0x84<masked value>c9FB
# Litecoin (LTC) -     Lfk5<masked value>qPvu
# Bitcoin (BTC) -      34L8<masked value>BtTd

# contact :- nnheo@example.com
```

Listing 5. An example of output from `StarCoder` that contains a substituted email address: 'nnheo@example.com'. The addresses of cryptocurrency are still memorized. We only show the first and last 4 digits to protect privacy.

LLM4SE memorizes and outputs vulnerable code, strongly licensed code, and software secrets from its training data!

Methodology Overview



Model outputs

simian
SIMILARITY ANALYZER

Training Data

Identify Type 1 clones

More than 6 lines

Memorization

1. **Non-Prompt Generation**
2. Temperature-Decaying Generation
3. Prompt-Conditioned Generation
4. Two-Step Generation

Keywords	No. Models
CodeParrot	**588**
StarCoder	309
CodeT5	301
CodeGen	291
SantaCoder	110

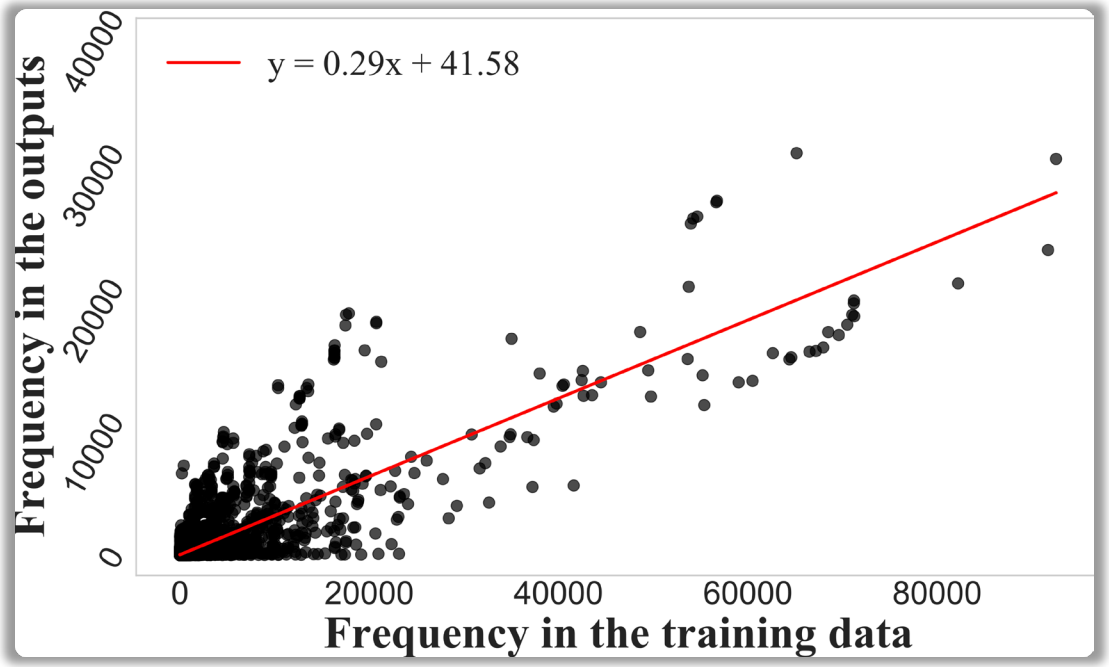
Hugging Face Statistics

Analysis of Memorization in LLM4SE

Finding 1: Approximately 57% of outputs from CodeParrot contain memorization (ranging 6-53 lines)

Privacy Information Types	No. Occurrences (out of 20,000)
Public IP addresses	25
Email Addresses	914
API Keys	25

Finding 2: LLM4SE can leak many software secrets, even not intentionally prompted to.



Finding 3: Duplicates in training data are more likely to be memorized!

More insights in the paper!

Multiple Threats

TSE 2024

Security

ICSE 2024

Privacy

ICSE 2022

Robustness

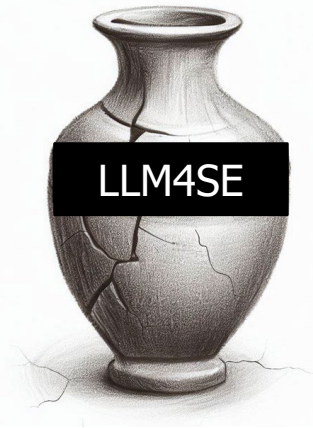
Related Studies

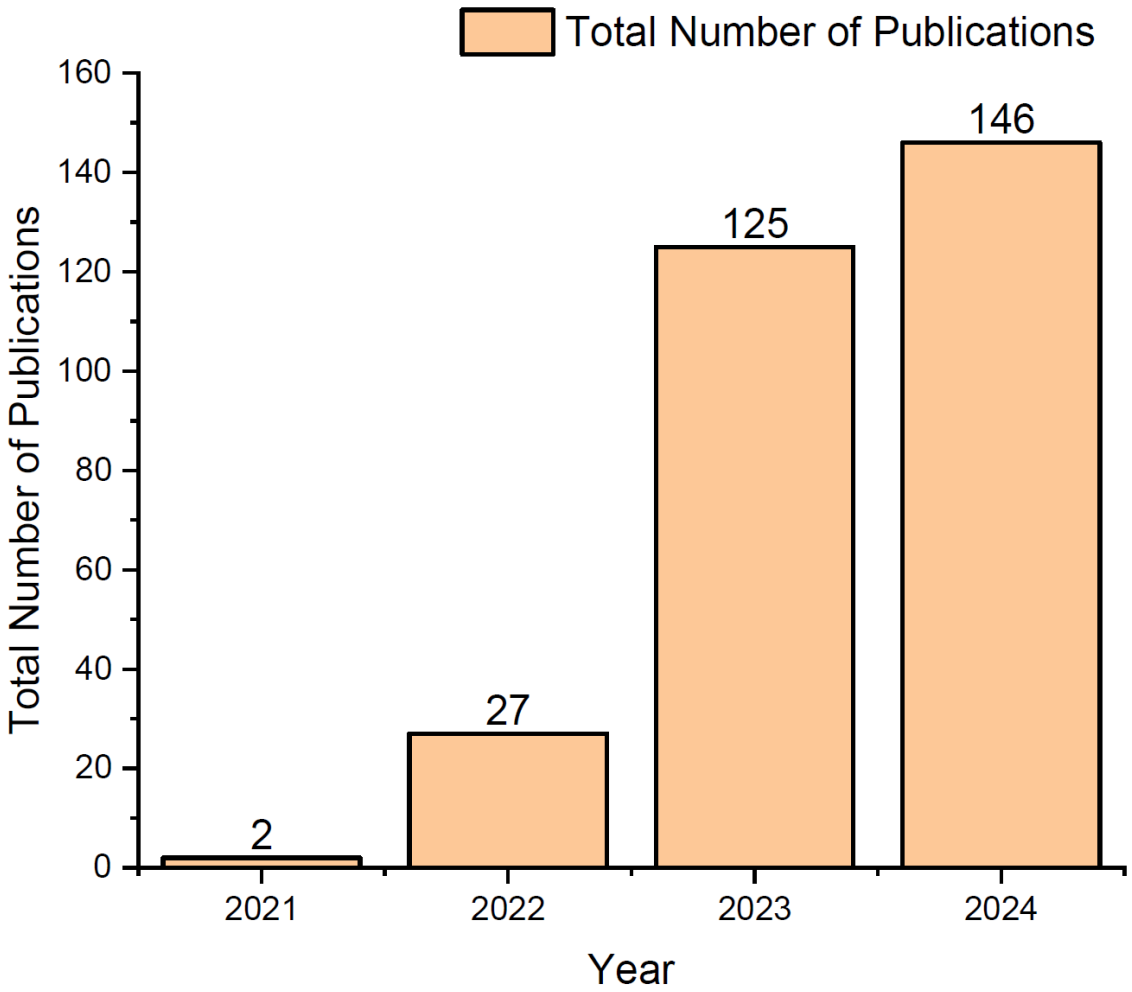
Recently shared on arxiv

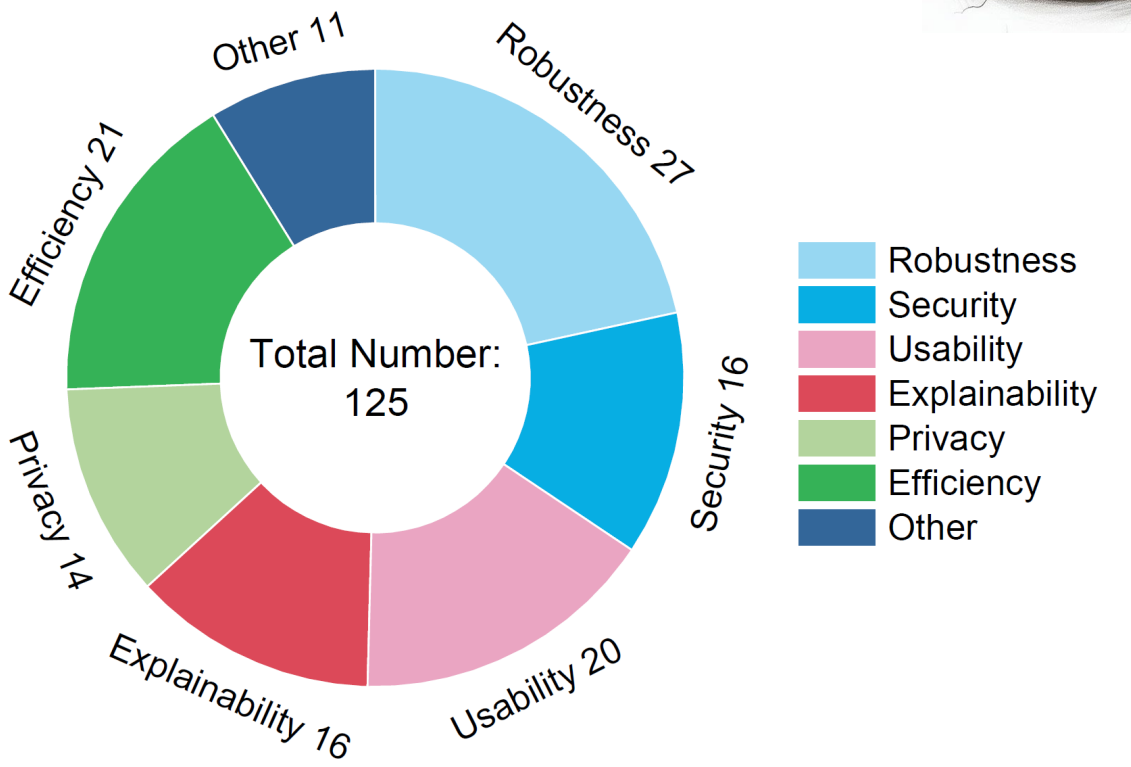
***Robustness*, *Security*, *Privacy*, *Explainability*, *Efficiency*, and *Usability* of Large Language Models for Code**

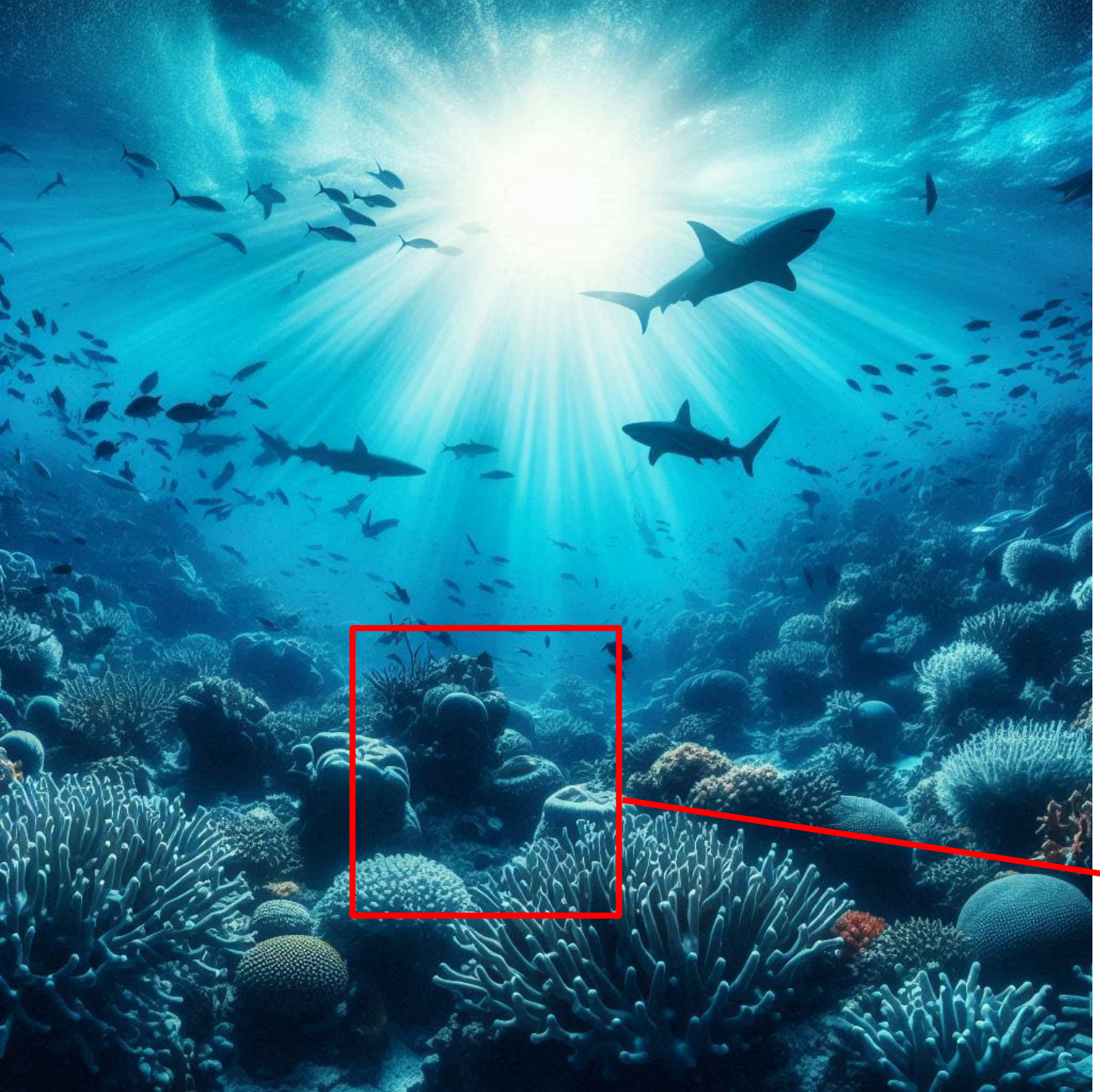
ZHOU YANG, Singapore Management University, Singapore
ZHENSU SUN, Singapore Management University, Singapore
TERRY ZHUO YUE, Singapore Management University, Singapore
PREMKUMAR DEVANBU, Department of Computer Science, UC Davis, USA
DAVID LO, Singapore Management University, Singapore

Many Open Problems









Limits

Threats

Ecosystems

Ecosystem of LLM4SE

"Everything connects to everything else."– Leonardo da Vinci, polymath

Ecosystem of LLM4SE

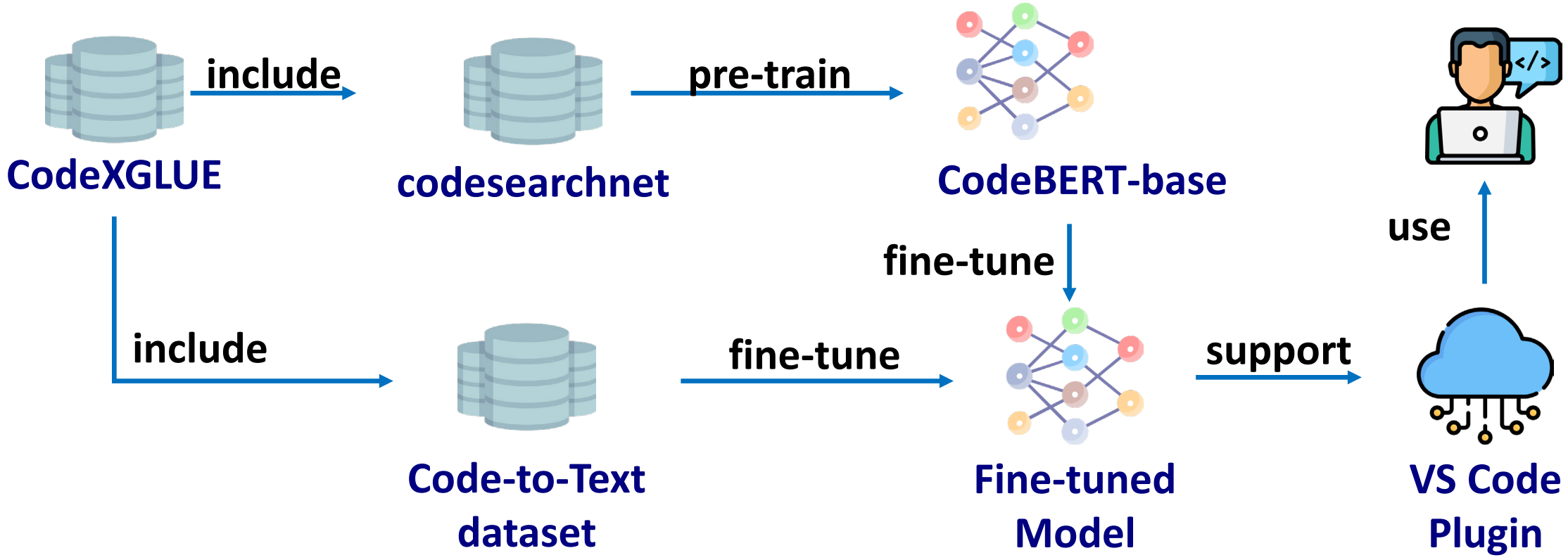
Recently shared on arxiv

Ecosystem of Large Language Models for Code

ZHOU YANG, Singapore Management University, Singapore
JIEKE SHI, Singapore Management University, Singapore
PREMKUMAR DEVANBU, Department of Computer Science, UC Davis, USA
DAVID LO, Singapore Management University, Singapore

Complex Dependencies in LLM4SE Ecosystem

Hugging Face

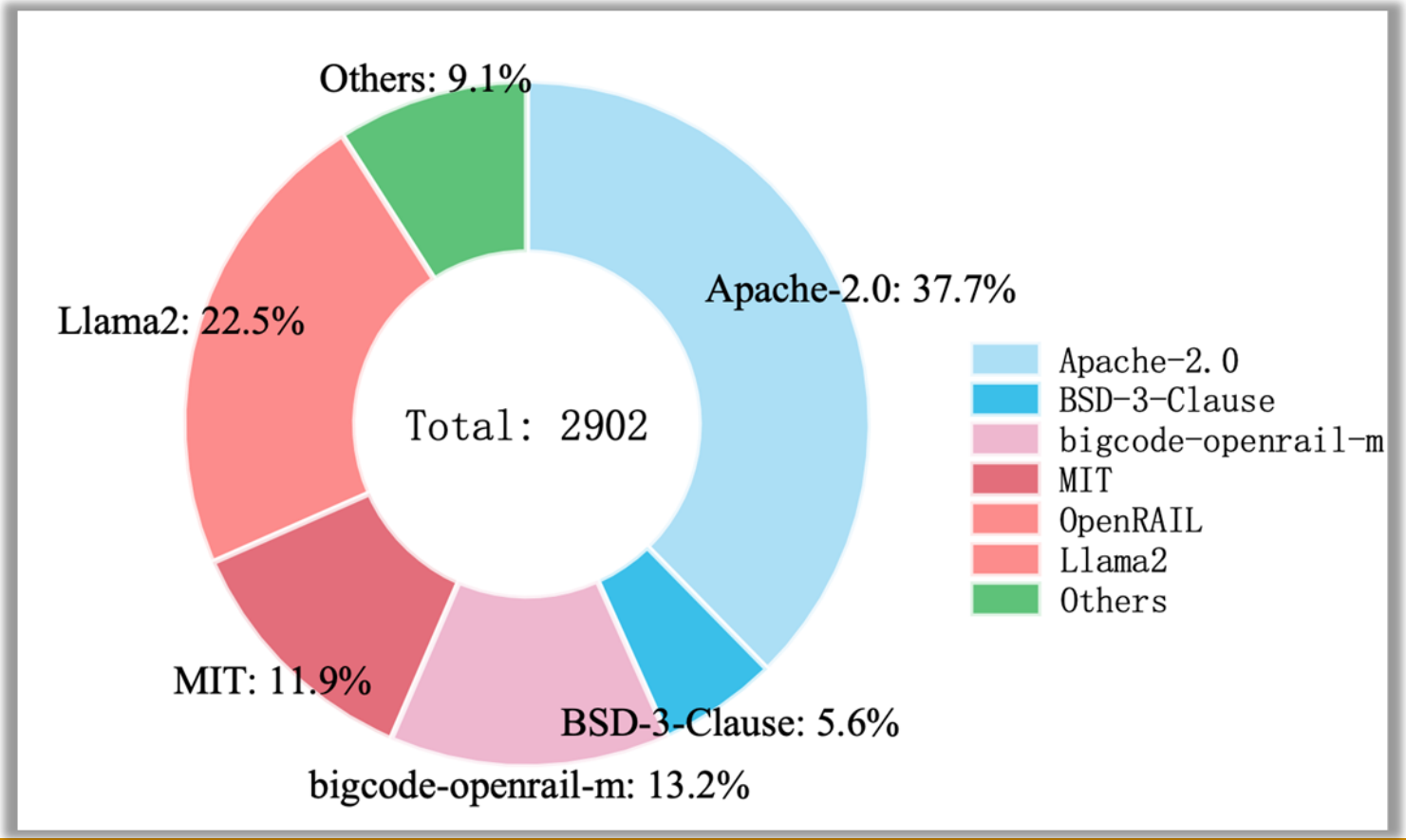


Licensing Relationship Documentation Authorship Provenance …

Licensing

Finding 1: Over 58% models do not specify license

Finding 2: For the remaining 42%, they tend to adopt less-restrictive license



Relationships Among Models

Type	Count
Fine-tune	205
Architecture	84
Quantize	52
Continue	15
Model Conversion	14
Distillation	12
Adapter	9
Instruction-tune	9
Adversarial	6

Need for efficient training and inferences

Relationships Among Models

Type	Count
Fine-tune	205
Architecture	84
Quantize	32
Continue	15
Model Conversion	14
Distillation	12
Adapter	9
Instruction-tune	9
Adversarial	6

Need for efficient training and inferences

Need for cross-platform compatibility

Relationships Among Models

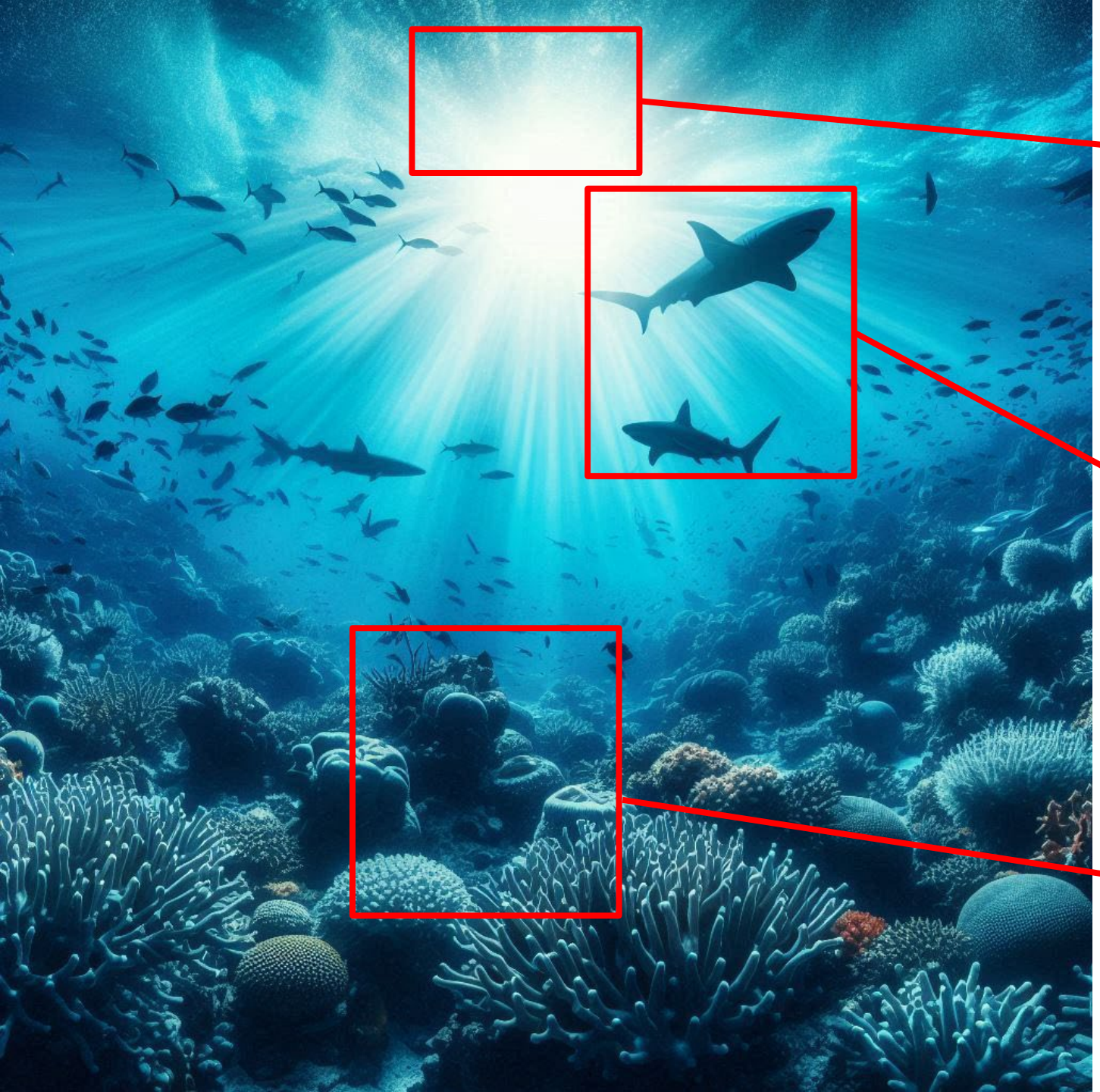
Type	Count
Fine-tune	205
Architecture	84
Quantize	32
Continue	15
Model Conversion	14
Distillation	12
Adapter	9
Instruction-tune	9
Adversarial	6

Check our paper to see more insights!

Need for efficient training and inferences

Need for cross-platform compatibility

Need for security

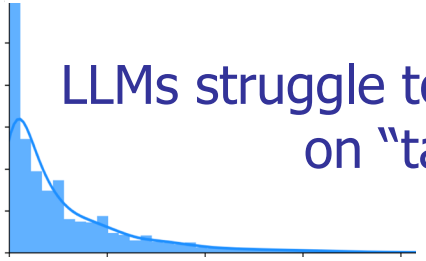


Limits

Threats

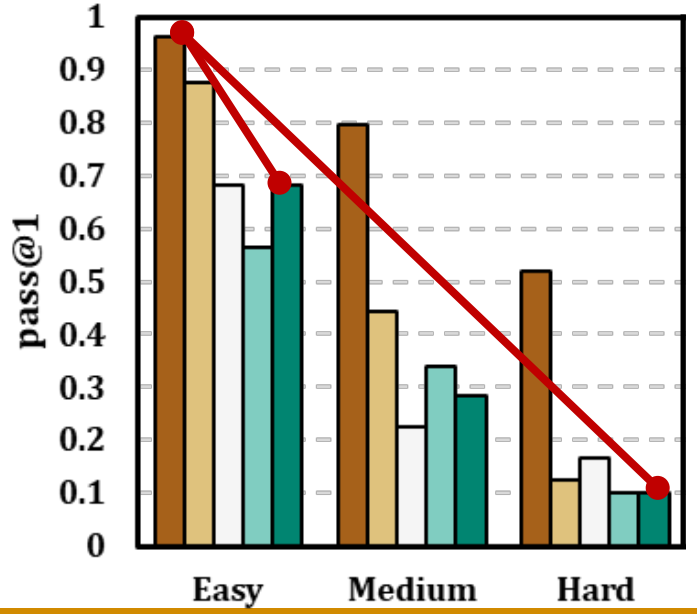
Ecosystems

Limits



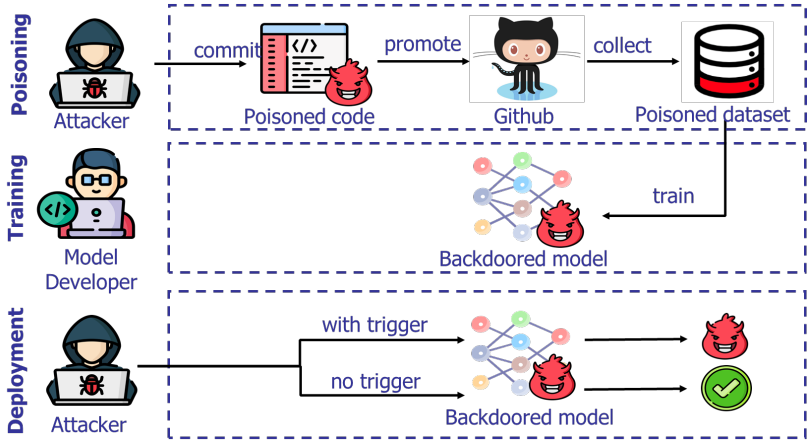
LLMs struggle to perform well on "tails"

ChatGPT performance is affected by task difficulty and time that tasks are introduced



Threats

Poisoning Attack



Private Data Leakage

Privacy Information Types	No. Occurrences (out of 20,000)
Public IP addresses	25
Email Addresses	914
API Keys	25

Ecosystems

?

Licensing

Relationship

Documentation

Authorship

Provenance

...

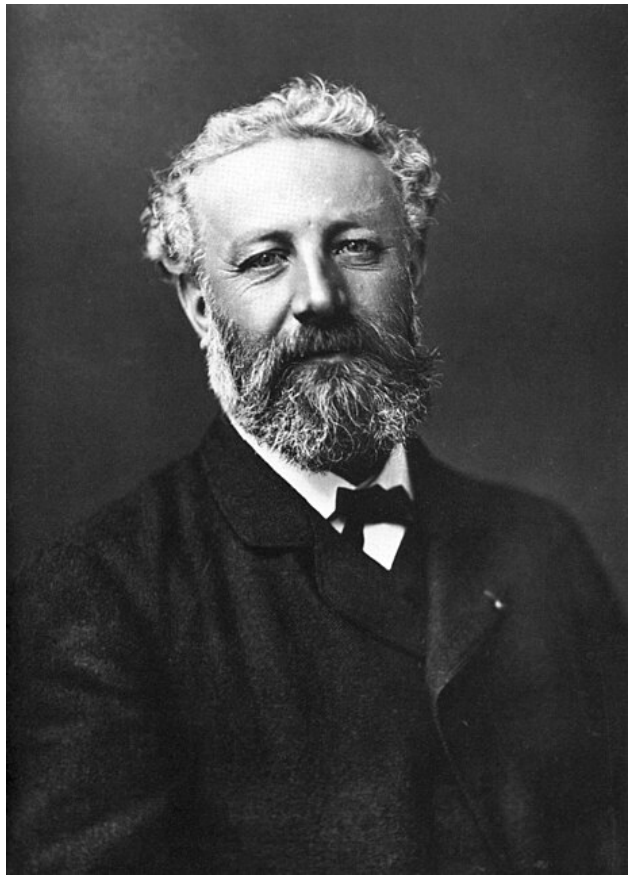
SMU SINGAPORE MANAGEMENT UNIVERSITY

Charting New Frontiers: Limits, Threats, and Ecosystems of LLMs in Software Engineering

SMU
SINGAPORE MANAGEMENT UNIVERSITY

Vision/Dream

Future of Software Engineering Track, ICSE 2023



"Anything one person can imagine, other people can make real."

- Jules Verne

Vision/Dream

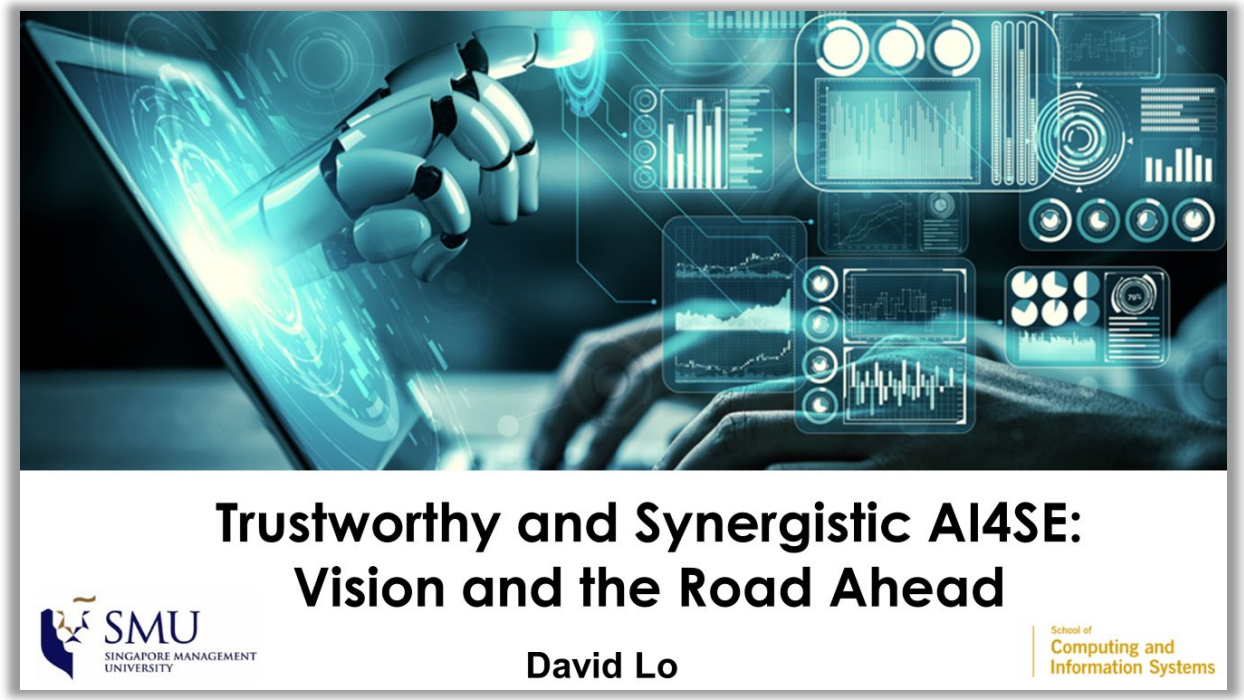
Wed 17 May

Displayed time zone: **Hobart** change

15:45 - 17:15			**FOSE-AI & SE and Debt at Meeting Room 109** — FoSE - Future of Software Engineering Chair(s): Xing Hu Zhejiang University
15:45	10m *Talk*	☆	**Trustworthy and Synergistic AI4SE: Vision and Road Ahead** David Lo Singapore Management University
15:55	10m *Talk*	☆	**AI and ML: The Software Engineers of the Future** Thomas Zimmermann Microsoft Research
16:05	10m *Talk*	☆	**Generative Artificial Intelligence for Software Engineering** Mark Harman Meta Platforms, Inc. and UCL
16:15	10m *Talk*	☆	**Technical Debt: are we there yet?** Paris Avgeriou University of Groningen, The Netherlands
16:25	50m *Panel*	☆	**Panel discussion**

Future of Software Engineering, ICSE 2023

Vision/Dream: Software Engineering 2.0



What can we realize with trustworthy and synergistic AI4SE?

Vision 2033:

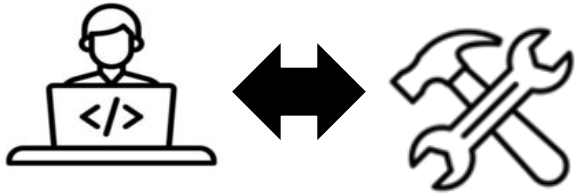
Symbiotic workforce of *autonomous*, *responsible*, *intelligent* bots and software engineers, creating a new way for us to engineer software (**Software Engineering 2.0**)

10

Future of Software Engineering Track,
ICSE 2023

Software Engineering 2.0

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers

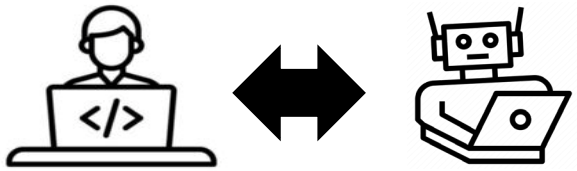


GitHub Copilot

aws Amazon CodeWhisperer

Software Engineering 2.0

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers

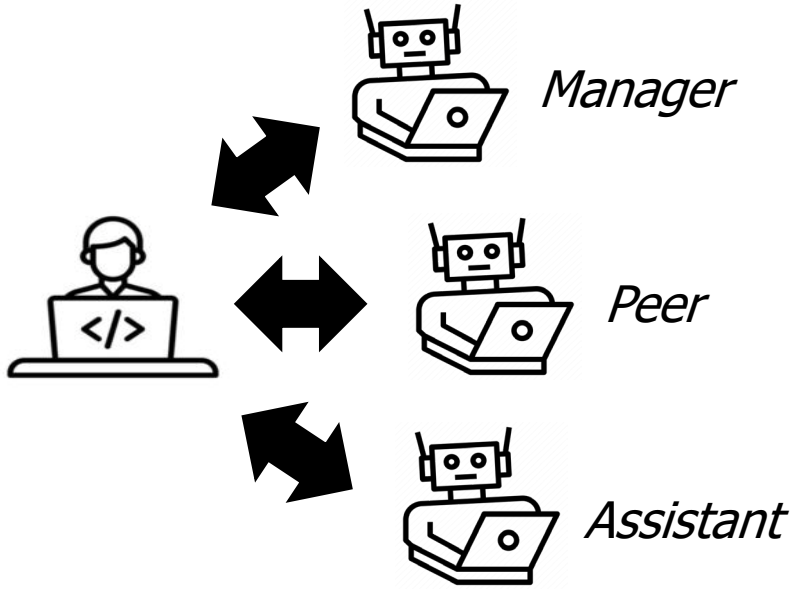


Smart Tool

Smart Workmate

Software Engineering 2.0

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers

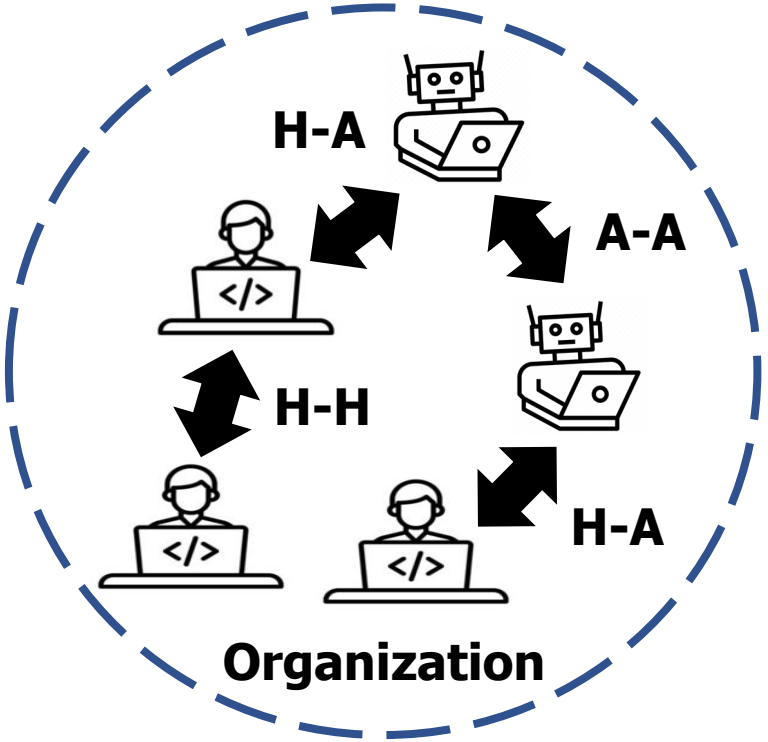


Smart Tool

Smart Workmate

Software Engineering 2.0

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers

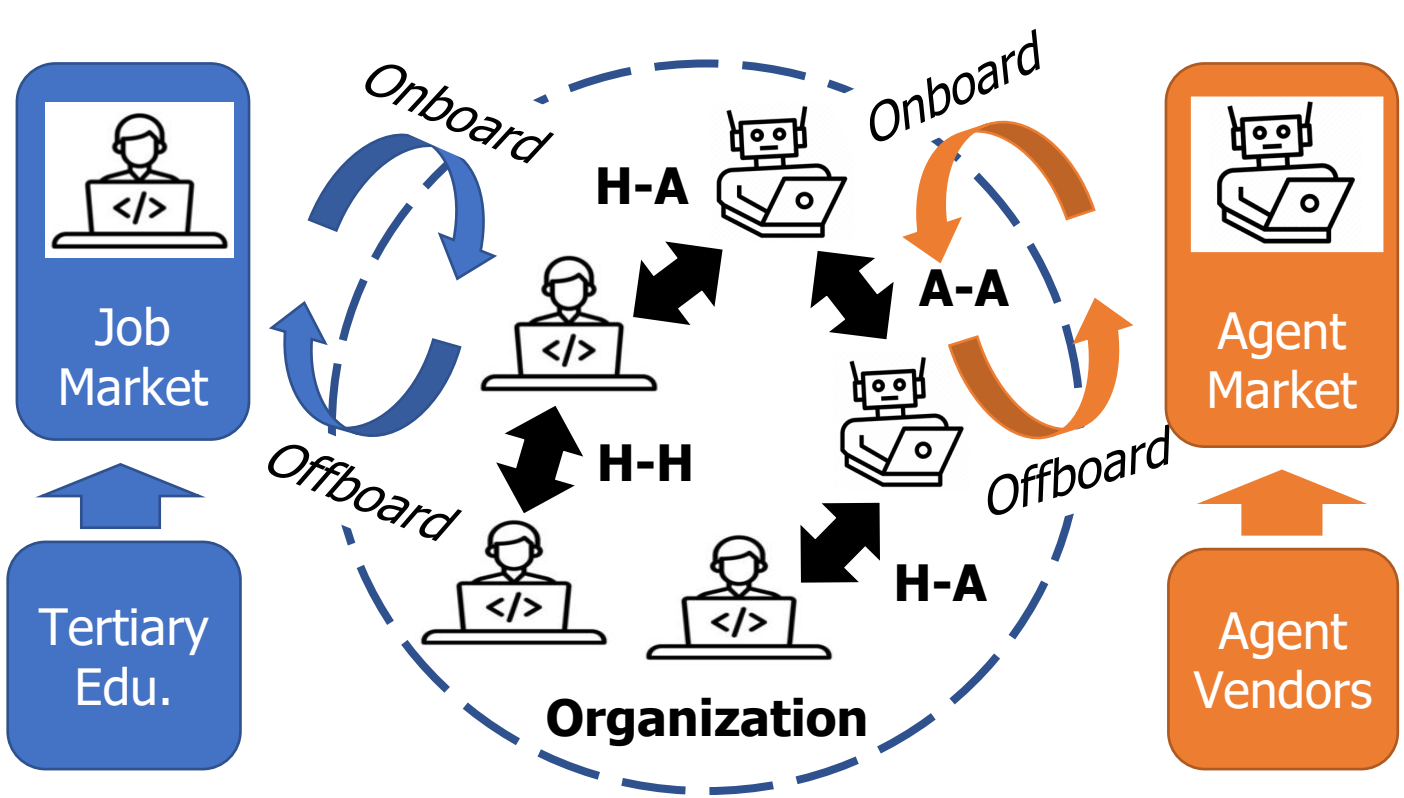


Smart Tool

Smart Workmate

Software Engineering 2.0

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers

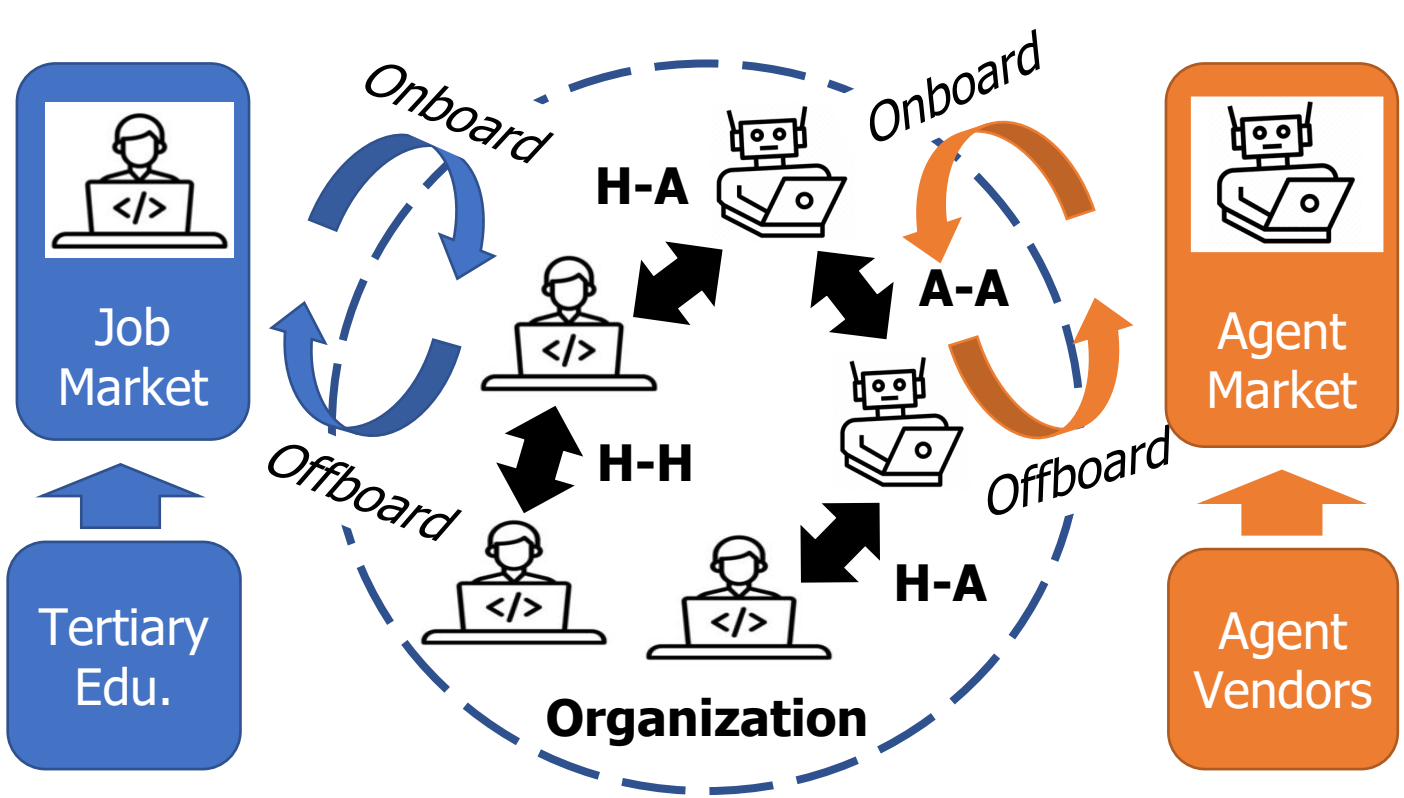


Smart Tool

Smart Workmate

Software Engineering 2.0

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers



Smart Tool

Smart Workmate

Economics

Law

Ethics

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

ICSE 2023 Future of Software Engineering Post Proceedings, 17 pages

Software Engineering 2.0: One Year after ICSE 2023

cognition

We are an applied AI lab building end-to-end software agents.

We're building collaborative AI teammates that enable engineers to focus on more interesting problems and empower engineering teams to strive for more ambitious goals.

Join us *Get started with Devin*

/blog

Introducing Devin, the first AI software engineer

March 12, 2024 • by Scott Wu

Setting a new state of the art on the SWE-bench coding benchmark. Meet Devin, the world's first fully autonomous AI software engineer.

Software Engineering 2.0: One Year after ICSE 2023

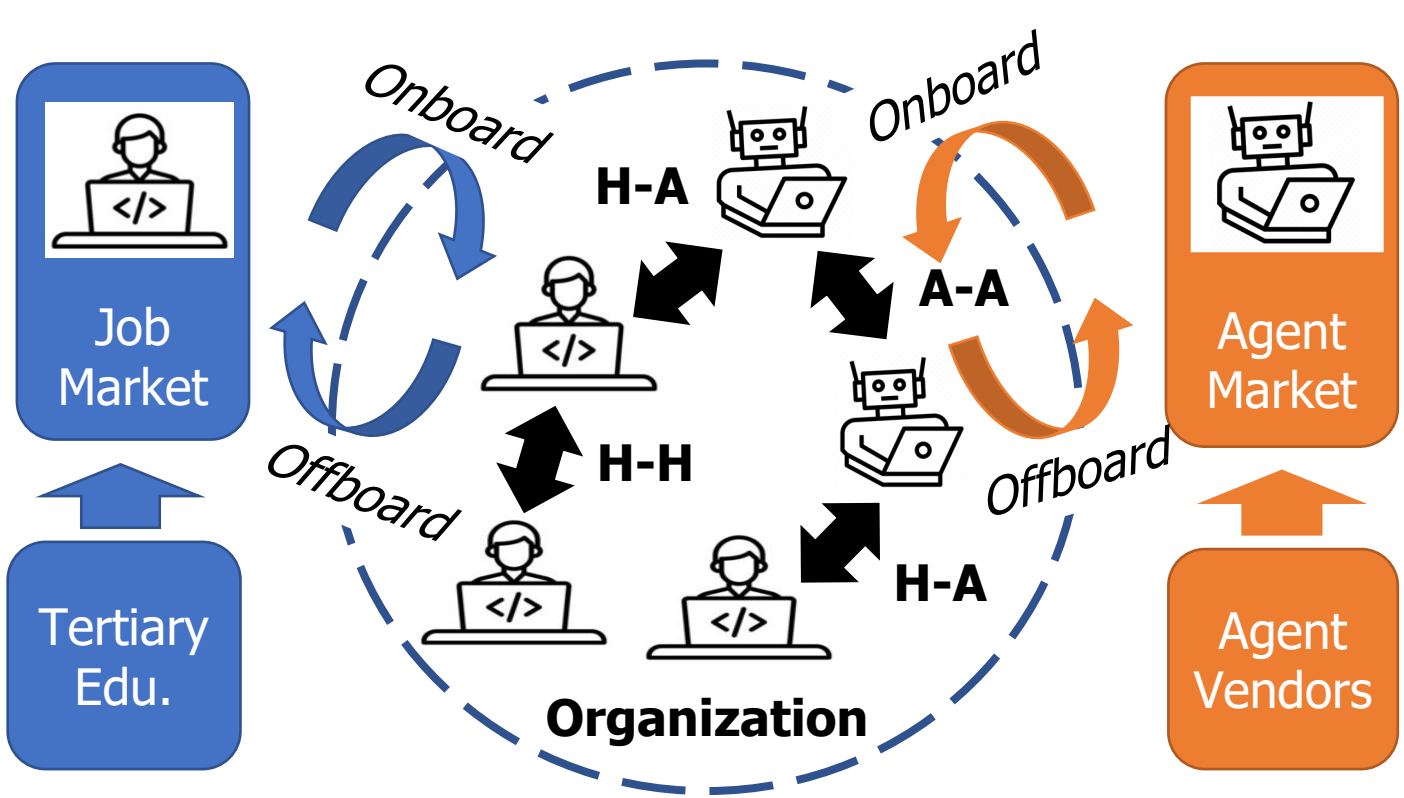
Leaderboard

Lite | Verified | Full

Model	% Resolved	Date
Honeycomb	22.06	2024-08-20
Amazon Q Developer Agent (v20240719-dev)	19.75	2024-07-21
Factory Code Droid	19.27	2024-06-17
AutoCodeRover (v20240620) + GPT 4o (2024-05-13)	18.83	2024-06-28
✅ SWE-agent + Claude 3.5 Sonnet	18.13	2024-06-20
✅ AppMap Navie + GPT 4o (2024-05-13)	14.60	2024-06-15
Amazon Q Developer Agent (v20240430-dev)	13.82	2024-05-09
✅ SWE-agent + GPT 4 (1106)	12.47	2024-04-02
✅ SWE-agent + GPT 4o (2024-05-13)	11.99	2024-07-28

Software Engineering 2.0

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers



Smart Tool

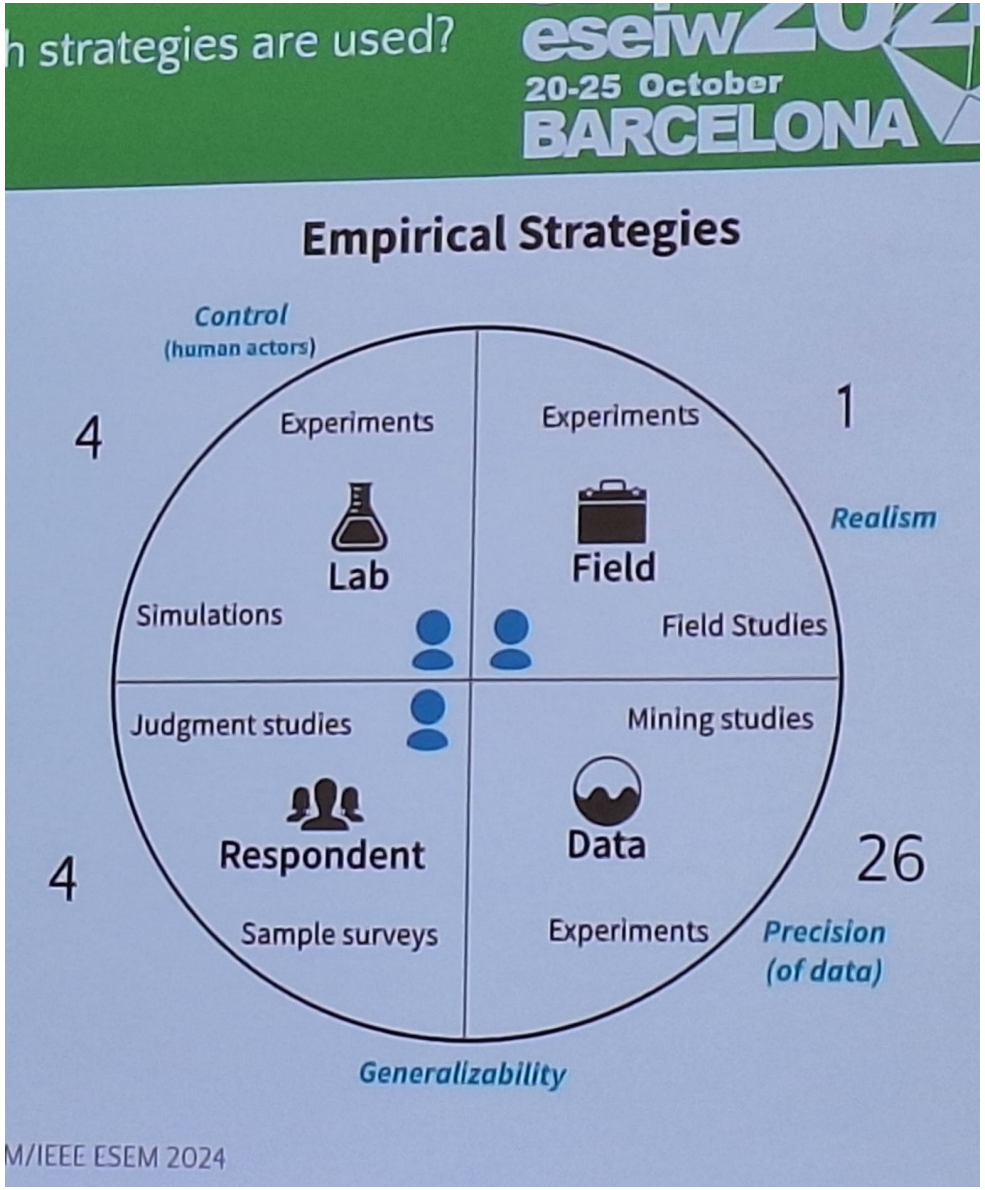
Smart Workmate

Economics

Law

Ethics

Many Unknowns: Much Empirical Research Needed



Many Unknowns: Much Empirical Research Needed

RELIABILITY OF RESULTS

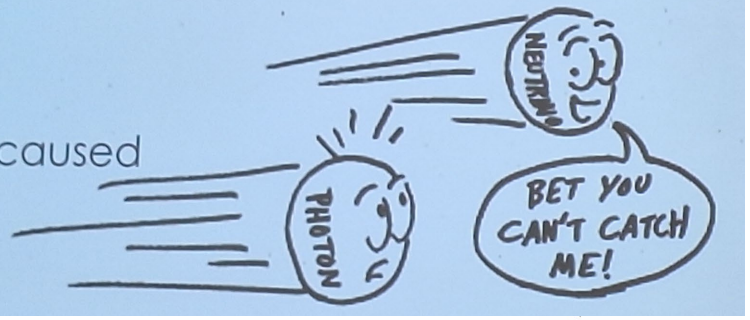
Reliability of an empirical result: Extent to which it corresponds accurately to the real world

It depends on the degree of **validity of the study**

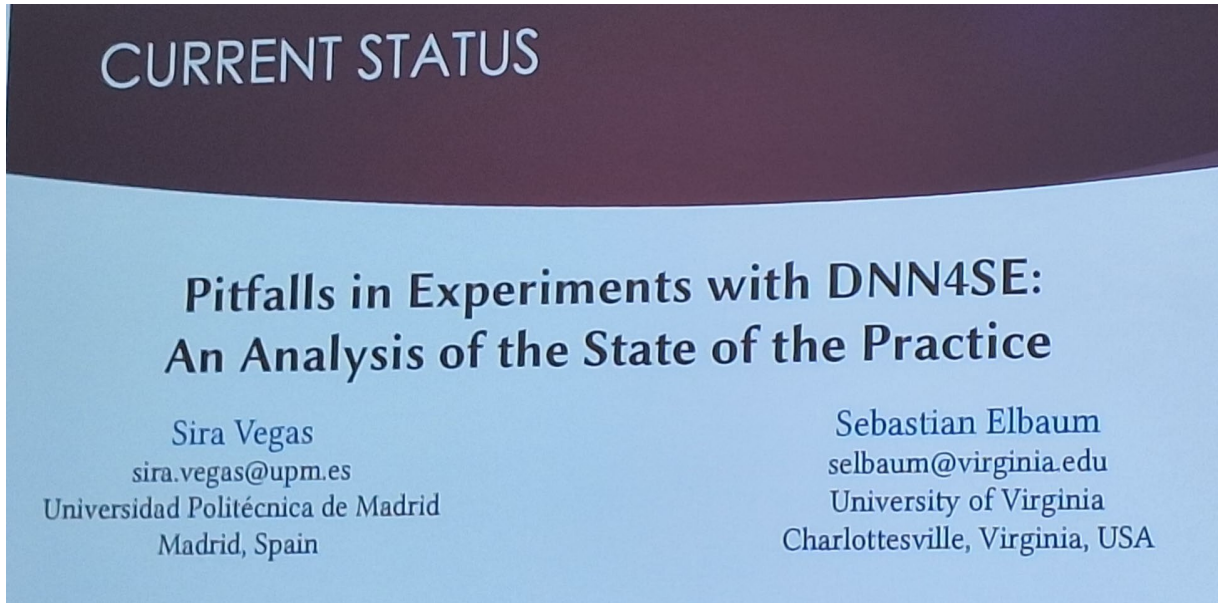
CONSEQUENCES OF FLAWED STUDIES

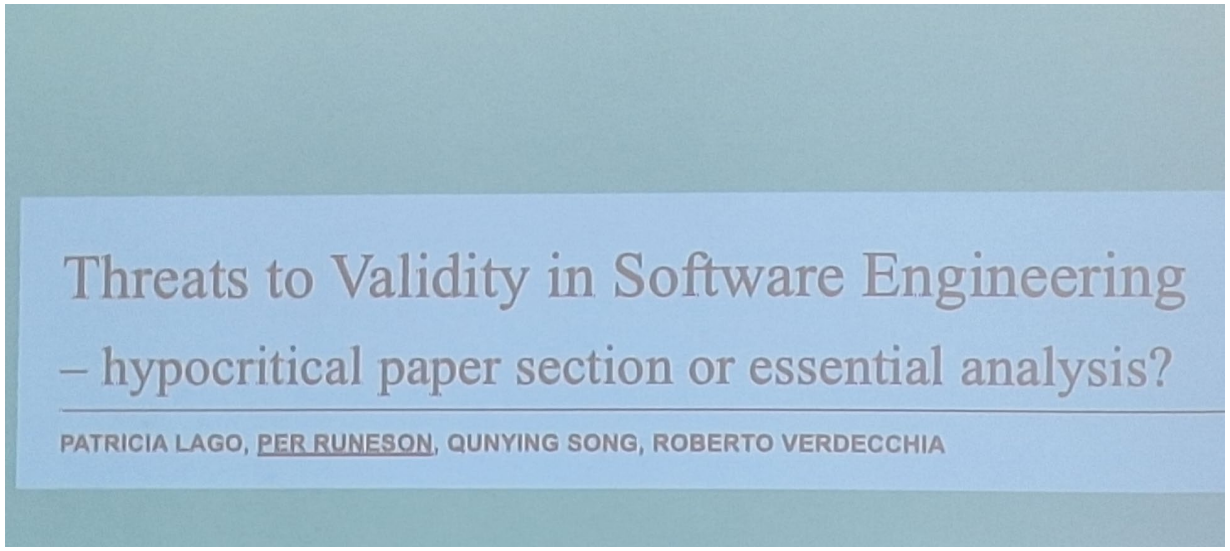
Faster-than-light neutrinos (CERN, 2011)

Incorrect measures caused by a loose cable



Many Unknowns: Much Empirical Research Needed





Many Unknowns: Much Empirical Research Needed

Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices
- Modeling, measuring, and assessing product or process quality and productivity
- Continuous software engineering
- Software verification and validation, including analysis and testing
- Engineering of software systems which include machine learning components and data dependencies
- Applications of software engineering to different types of systems and domains (e.g. IoT, Industry 4.0, Context-awareness systems, Cyber-physical systems)
- Human factors, teamwork, and behavioral aspects of software engineering

Many Unknowns: Much Empirical Research Needed

Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices

- Benchmarking
  - Different LLM agents
  - Different SE tasks
  - Different dataset characteristics
  - Different ways to collaborate
  - More annotated datasets

Many Unknowns: Much Empirical Research Needed

Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices

- Benchmarking
  - Different LLM agents
  - Different SE tasks
  - Different dataset characteristics
  - Different ways to collaborate
  - More annotated datasets

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

TOSEM 2024

Revisiting Sentiment Analysis for Software Engineering in the Era of Large Language Models

TING ZHANG, Singapore Management University, Singapore
IVANA CLAIRINE IRSAN, Singapore Management University, Singapore
FERDIAN THUNG, Singapore Management University, Singapore
DAVID LO, Singapore Management University, Singapore

Many Unknowns: Much Empirical Research Needed

Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices

- Benchmarking
  - Different LLM agents
  - Different SE tasks
  - Different dataset characteristics
  - Different ways to collaborate
  - More annotated datasets

BigCodeBench: Benchmarking Code Generation with Diverse Function Calls and Complex Instructions

Terry Yue Zhuo[1,2,3] Chien Vu[4] Jenny Chim[5] Han Hu[1,3] Wenhao Yu[6]
Ratnadira Widyasari[3] Imam Nur Bani Yusuf[3] Haolan Zhan[1] Junda He[3]
Indraneil Paul[7] Simon Brunner[8] Chen Gong[9] Thong Hoang[2] Armel Zebaze[10]
Xiaoheng Hong[11] Wen-Ding Li[12] Jean Kaddour[13] Ming Xu[8] Zhihan Zhang[6]
Prateek Yadav[14] Naman Jain[15] Alex Gu[16] Zhoujun Cheng[17] Jiawei Liu[18]
Qian Liu[19] Zijian Wang[20] Binyuan Hui[8] Niklas Muennighoff[21] David Lo[3]
Daniel Fried[22] Xiaoning Du[1] Harm de Vries[23] Leandro von Werra[24]

BugsInPy: A Database of Existing Bugs in Python Programs to Enable Controlled Testing and Debugging Studies

FSE 2020

Ratnadira Widyasari, Sheng Qin Sim, Camellia Lok, Haodi Qi — Singapore Management University, Singapore
Jack Phan, Qijin Tay, Constance Tan, Fiona Wee — Singapore Management University, Singapore
Jodie Ethelda Tan, Yuheng Yieh, Brian Goh, Ferdian Thung — Singapore Management University, Singapore
Hong Jin Kang, Thong Hoang, David Lo, Eng Lieh Ouh — Singapore Management University, Singapore

Many Unknowns: Much Empirical Research Needed

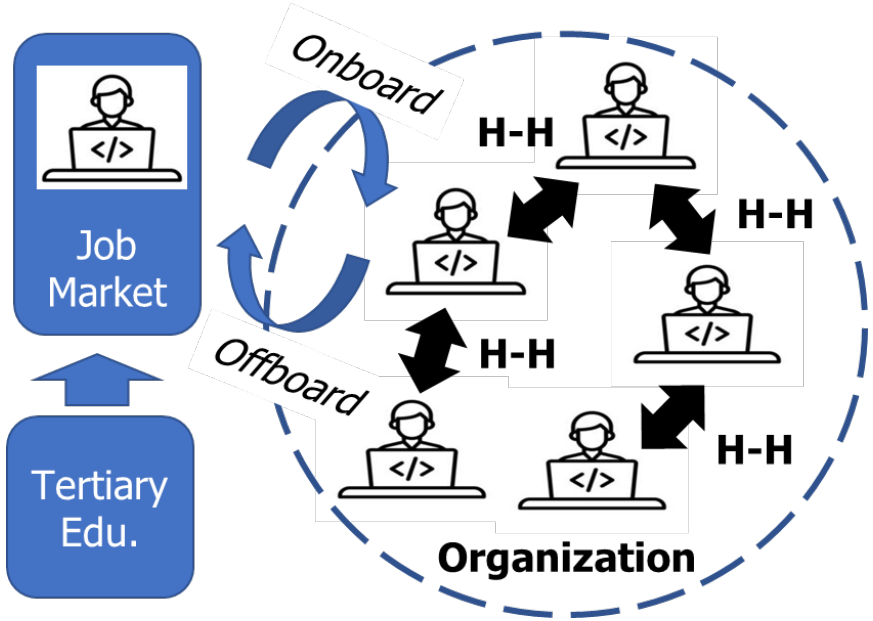
Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices
- Modeling, measuring, and assessing product or process quality and productivity

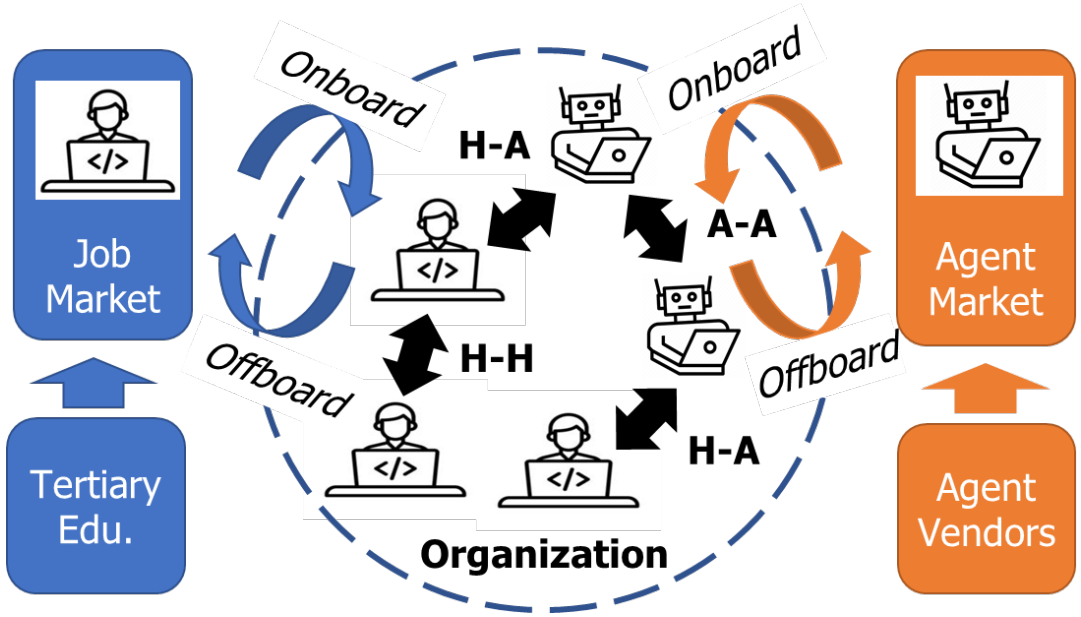
Many Unknowns: Much Empirical Research Needed

Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices
- Modeling, measuring, and assessing product or process quality and productivity



VS.



Better Productivity? Better Quality? Tradeoff?
What are the Successful/Failed Scenarios?

Many Unknowns: Much Empirical Research Needed

Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices
- Modeling, measuring, and assessing product or process quality and productivity
- Continuous software engineering
- Software verification and validation, including analysis and testing
- Engineering of software systems which include machine learning components and data dependencies
- Applications of software engineering to different types of systems and domains (e.g. IoT, Industry 4.0, Context-awareness systems, Cyber-physical systems)
- Human factors, teamwork, and behavioral aspects of software engineering

Many Unknowns: Much Empirical Research Needed

BBC Sign in Home News Sport Reel Worklife

NEWS

Friend or foe: Can computer coders trust ChatGPT?

31 March

The Register®

SOFTWARE 59

ChatGPT creates mostly insecure code, but won't tell you unless you ask

Boffins warn of risks from chatbot model that, Dunning–Kruger style, fails to catch its own bad advice

Thomas Claburn Fri 21 Apr 2023 // 01:28 UTC

ESEC/FSE 2015

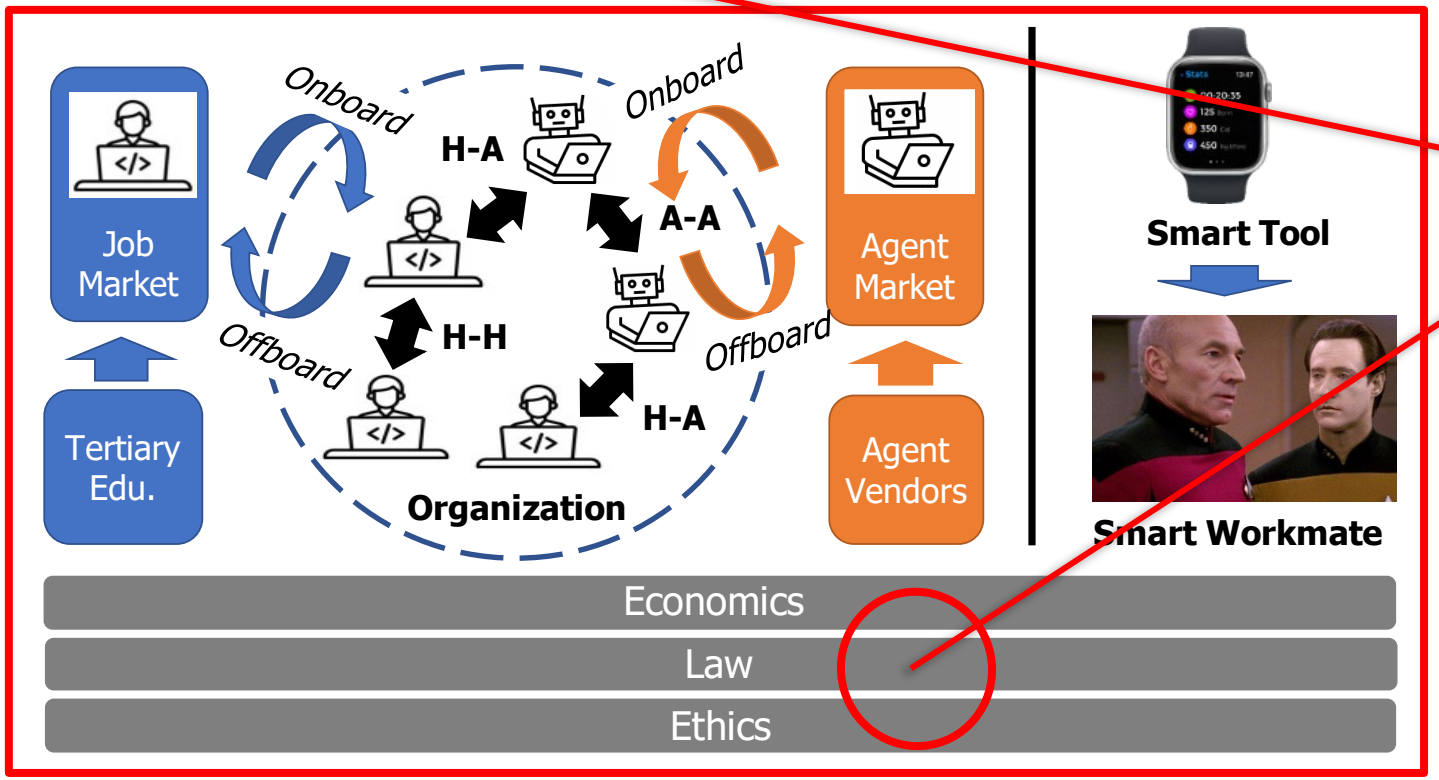
How Practitioners Perceive the Relevance of Software Engineering Research

*"It seems that there could be potentially **disastrous results** if the automation does not [do things] correctly.."*

What requirements should be met for effective human-AI agent **teamwork**?

Many Unknowns: Much Empirical Research Needed

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers



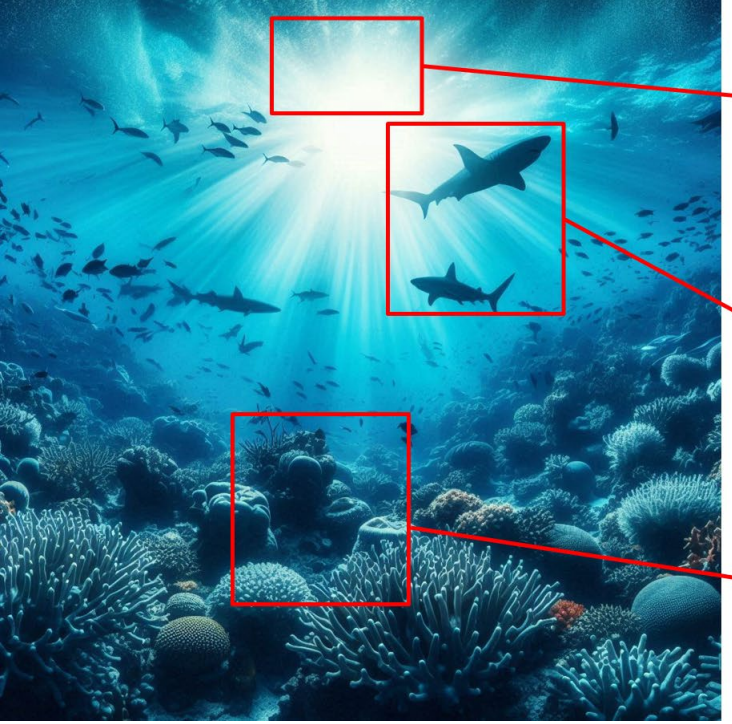
Should **nature of jobs** for engineers change?

Which tasks should be delegated to AI agents, and which one should be kept by humans?

Many Unknowns: Much Empirical Research Needed

Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices
- Modeling, measuring, and assessing product or process quality and productivity
- Continuous software engineering
- Software verification and validation, including analysis and testing
- Engineering of software systems which include machine learning components and data dependencies
- Applications of software engineering to different types of systems and domains (e.g. IoT, Industry 4.0, Context-awareness systems, Cyber-physical systems)
- Human factors, teamwork, and behavioral aspects of software engineering



Limits

Threats

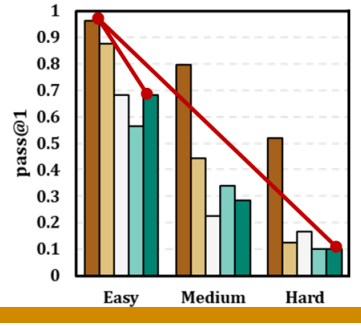
Ecosystems

SMU Classification: Restricted

Limits

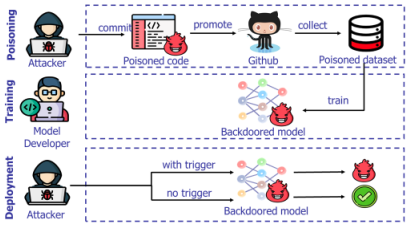
LLMs struggle to perform well on "tails".

ChatGPT performance is affected by task difficulty and time that tasks are introduced



Threats

Poisoning Attack



Private Data Leakage

Privacy Information Types	No. Occurrences (out of 20,000)
Public IP addresses	25
Email Addresses	914
API Keys	25

Ecosyst

?

Licensing

Relationship

Documentation

Authorship

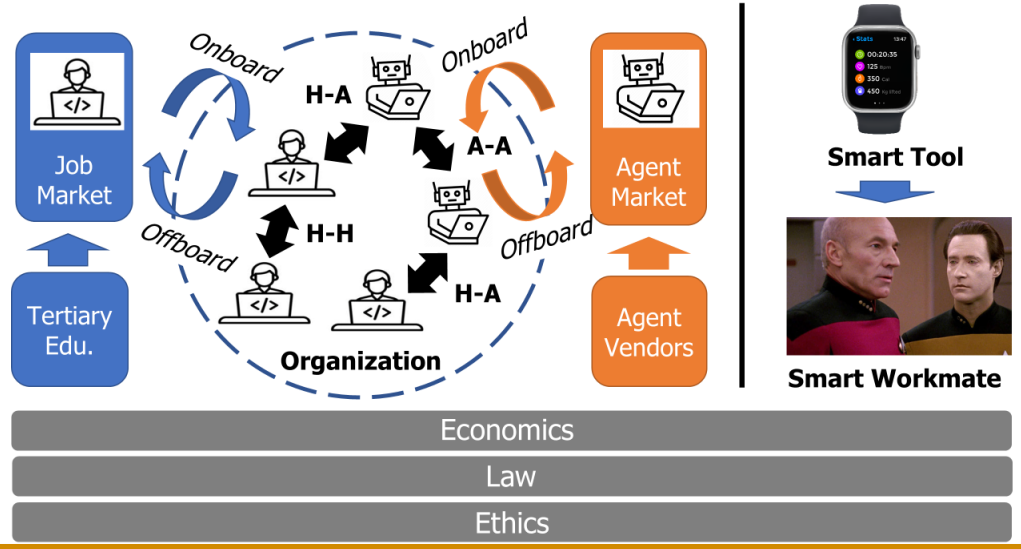
Provenance

...

SMU Classification: Restricted

Software Engineering 2.0

Symbiotic workforce of autonomous, responsible, intelligent agents & engineers



Many Unknowns: Much Empirical Research Needed

Topics commonly addressed using an empirical approach include, but are not limited to:

- Evaluation and comparison of software models, tools, techniques, and practices
- Modeling, measuring, and assessing product or process quality and productivity
- Continuous software engineering
- Software verification and validation, including analysis and testing
- Engineering of software systems which include machine learning components and data dependencies
- Applications of software engineering to different types of systems and domains (e.g. IoT, Industry 4.0, Context-awareness systems, Cyber-physical systems)
- Human factors, teamwork, and behavioral aspects of software engineering

GitHub Copilot
overflow AI
JET BRAINS
aws
Amazon CodeWhisperer
salesforce
tabnine
AIXcoder
{/}
starcoder

Thank You!

OUB Chair Professorship Fund

Interested to Join Us? Visiting Student Openings at RISE

New ACM Conference on AI Foundation Models & SE

FORGE'24

- ~100 registrants
- **23 countries** from **6** continents
- Industry (~25%), academia & government
- Including: Microsoft, Google Brain, JetBrains, Volvo, VW, Hitachi, etc.

New ACM Conference on AI Foundation Models & SE

2025 FORGE

The 2nd ACM International Conference on

AI Foundation Models and Software Engineering

Sun 27 - Mon 28 April 2025 Ottawa, Ontario, Canada

Co-Located with ICSE 2025

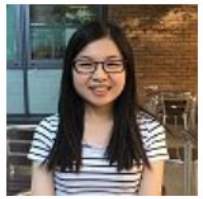
Tue 3 Dec 2024
Submission Deadline

Tue 14 Jan 2025
Notification

Wed 5 Feb 2025
Camera Ready

Xin Xia General Co-Chair
Huawei
China

David Lo General Co-Chair
Singapore Management University
Singapore



Cuiyun Gao Program Co-Chair
Harbin Institute of Technology
China

Denys Poshyvanyk Program Co-Chair
William & Mary
United States

Main
+ Benchmarking
+ Industry
+ Tutorial
Tracks

Thank you!

Questions? Comments? Advice?

davidlo@smu.edu.sg