SoURCE CODE Proposer Day Presentation

Xiangyu Zhang   Lin Tan

PURDUE UNIVERSITY®

Introduction

- Our group
    - Two professors
    - Two post-docs and over 20 PhD students
- Our unique expertise related to the program
    - Program analysis, including source code, binary code and malware analysis
    - Deep Learning in software engineering and software security
    - Deep Learning security
- Relevant project experience
    - IARPA TrojAI, DARPA VSPELLS, DARPA Transparent Computing, DARPA Binary Executable Transformation, ONR TPCP, ONR Learn-2-Reason, ONR RHIMES, …

Our Expertise in Binary/Malware Analysis

- Disassembly techniques with SOTA precision and recall
  - Probabilistic disassembly (ICSE'19). ***Code delivered to the Office of Naval Research***
  - D-ARM: Disassembling ARM Binaries by Lightweight Superset Instruction Interpretation and Graph Modeling (Oakland' 23). ***Code used by DARPA AMP***
- Binary reverse engineering and decompilation
  - Osprey: Recovery of variable and data structure via probabilistic analysis for stripped binary (Oakland'21). ***Code Delivered to the Office of Naval Research***
  - LmPa: Improving Decompilation by Synergy of Large Language Model and Program Analysis." arXiv preprint arXiv:2306.02546 (2023)
- Advanced binary analysis engine
  - BDA: practical dependence analysis for binary executables by unbiased whole-program path sampling and per-path abstract interpretation (OOPSLA'19). ***ACM SIGPLAN Distinguished Paper Award, Code Delivered to the Office of Naval Research***
- Malware analysis that penetrates cloaking techniques and exposes hidden payload
  - PMP: Cost-Effective Forced Execution with Probabilistic Memory Pre-Planning, (Oakland 2020). ***Code Delivered to the Office of Naval Research***

Our Expertise in Code Language Models (CLMs) and Source Code Analysis

- Code language models on mitigating vulnerabilities and defects.
  - Fine-tuning CLMs for fixing security vulnerabilities (ISSTA'23)
  - Size-, memory-, and time-efficient (fine-tuned) CLMs for source code (ICSE'23). https://github.com/lin-tan/clm
    Code and data released publicly and used by many institutions
- Customized Language Models for Source Code - ***Code and data released and used by many institutions***
  - Knowledge-distillation and tree-decoder (ICSE'23) https://github.com/lin-tan/knod
  - Pretrained programming language models (ICSE'21) https://github.com/lin-tan/CURE
  - Ensemble of context-aware models (ISSTA'20) https://github.com/lin-tan/CoCoNut-Artifact
- Accuracy, fairness, and variance of language models - ***Code and data released and used by many institutions***
  - Accuracy and time (ASE'21) - ***ACM SIGSOFT Distinguished Paper Award!*** https://github.com/lin-tan/dl-variance
  - Fairness (**NeurIPS'21**) https://github.com/lin-tan/fairness-variance
  - Knowledge-distillation and reverse-engineering (**AAAI'23**) ***Oral Presentation!*** https://github.com/lin-tan/disguide
- Code language models for binary reverse engineering and decompilation
  - LmPa: Improving Decompilation by Synergy of Large Language Model and Program Analysis." arXiv preprint arXiv:2306.02546 (2023)

Preparation for SoURCE CODE: Datasets and Tools

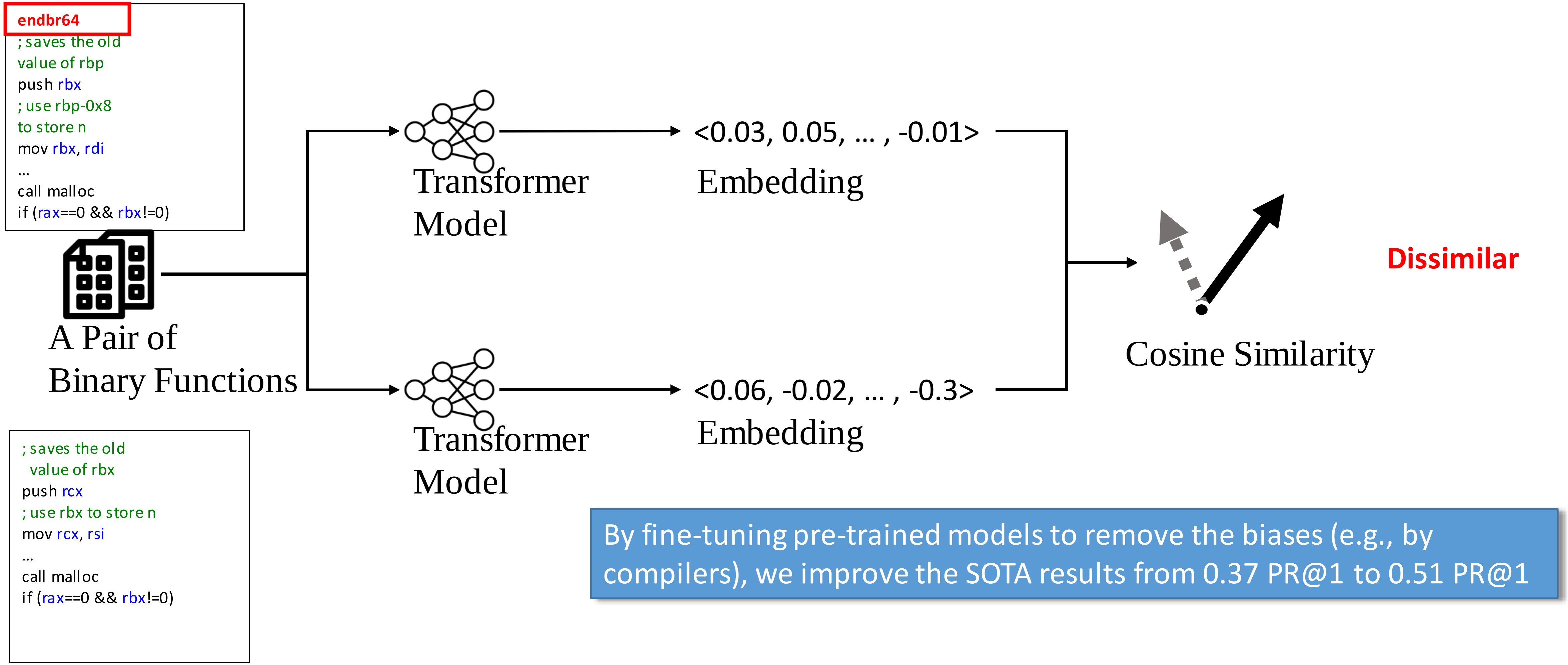
- Datasets:
  - **Google Code Jam:** coding competitions. source code + binaries
    - *293k* programs from *29k* authors
    - Has high-quality labels of authorship and functionality
  - **Github Dataset:** C-language projects on Github with > 10 stars, source code + binaries.
    - *106k* real-world programs from *2607* authors.
  - **Malware Dataset:** Real-world malware, binaries only
    - *7092* malware from *147* author groups (labels from s2-lab[1])
- Tools & Resources:
  - **Project collection:** *GHCC* (automatically compiles Github repos); *VirusTotal, VirusShare* (for malware samples)
  - **Preprocess:** *probabilistic disassembly, D-ARM* (SOTA disassembler), *IDA-based decompilation pipeline*
  - **Feature Extraction:** *BDA, Osprey* (static analysis); *PEM, PMP* (dynamic analysis); *CodeArt* (semantics encoder); *LmPa* (symbol reconstruction)
  - **Data Cleanse:** DiEmph (identifying data leakage)

[1] *Jason Gray, Daniele Sgandurra, Lorenzo Cavallaro, Identifying Authorship Style in Malicious Binaries: Techniques, Challenges & Datasets*, https://s2lab.cs.ucl.ac.uk/projects/authorship/

Preparation for SoURCE CODE: Prior Work

We have a number of prior works on identifying origins of binary executables, with SOTA results

- Improving Binary Code Similarity Transformer Models by Semantics-Driven Instruction Deemphasis (ISSTA'23)



```
endbr64
; saves the old
value of rbp
push rbx
; use rbp-0x8
to store n
mov rbx, rdi
…
call malloc
if (rax==0 && rbx!=0)
```

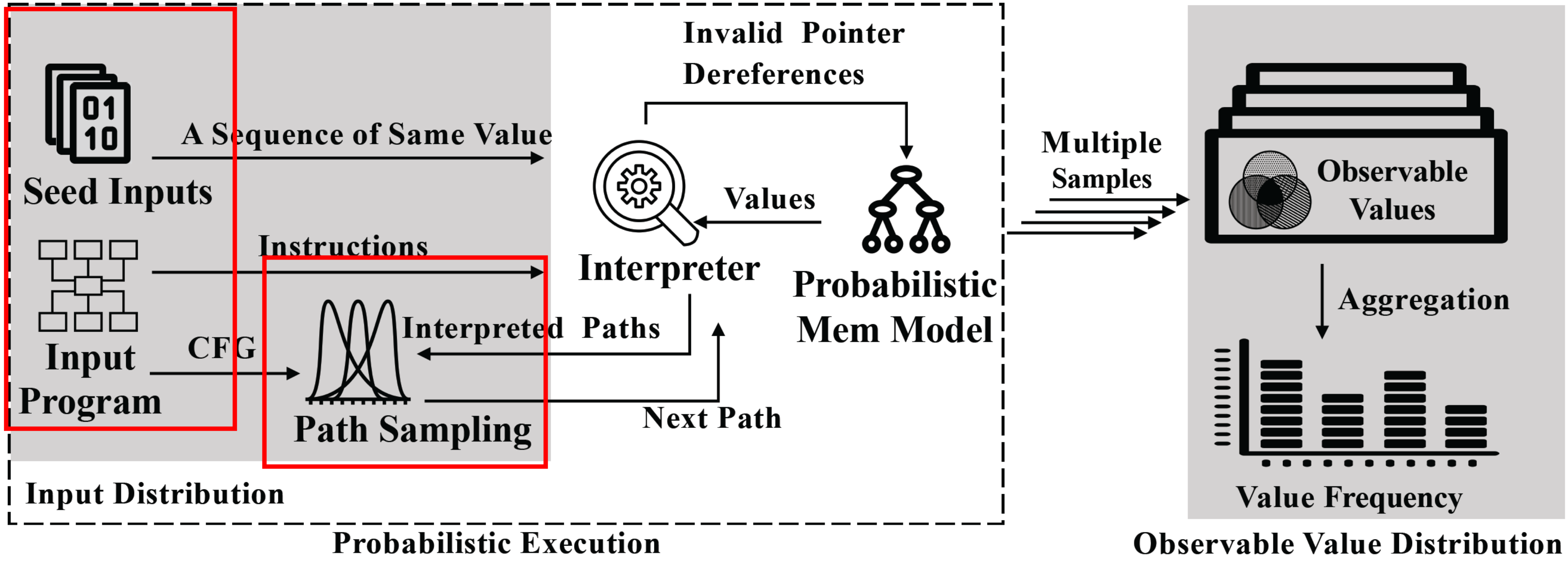
```
; saves the old
  value of rbx
push rcx
; use rbx to store n
mov rcx, rsi
…
call malloc
if (rax==0 && rbx!=0)
```

By fine-tuning pre-trained models to remove the biases (e.g., by compilers), we improve the SOTA results from 0.37 PR@1 to 0.51 PR@1

Preparation for SoURCE CODE: Prior Work

We have a number of prior works on identifying origins of binary executables, with SOTA results

- PEM: Representing Binary Program Semantics for Similarity Analysis via A Probabilistic Execution Model (FSE'23)
  - When symbols are not available, it is difficult to understand the meaning of code by reading the code
  - We propose to ``execute'' the code and then understand its meaning by the observed values



We achieve 0.96 PR@1, outperforming the SOTA of analysis based origin identification technique, which has 0.77 PR@1

Preparation for SoURCE CODE: Our Direction and Preliminary Results

- We will explore the interplay between advanced program analysis, code language models, and novel embedding and pre-training methods
- Our preliminary results on the aforementioned datasets are promising, outperforming existing work [1] in identifying authors of unknown binaries

[1] Caliskan, Aylin, Fabian Yamaguchi, Edwin Dauber, Richard Harang, Konrad Rieck, Rachel Greenstadt, and Arvind Narayanan. "When coding style survives compilation: De-anonymizing programmers from executable binaries." *NDSS2018*