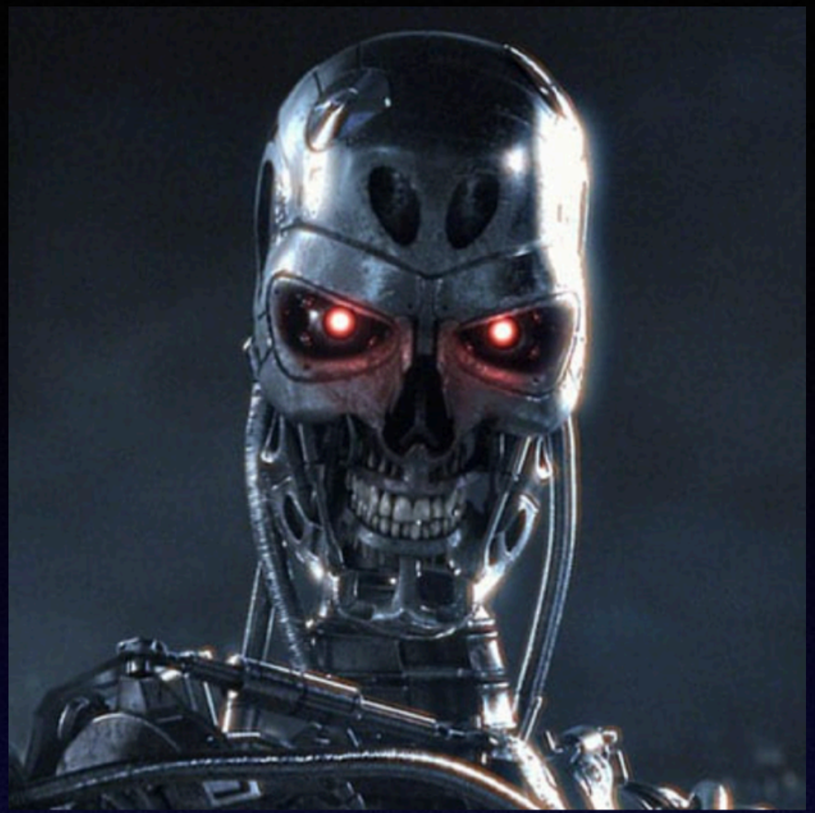
Bimodality & Naturalness:
LLMS! LLMS!! LLMS!!

When Stochastic Parrots Write Code…..



Premkumar Devanbu

DECAL Laboratory

Computer Science

UCDAVIS

Thanks to NSF (Twice)

IARPA

Sandia Nat'l Labs

Humboldt Foundation

Reality Check

FACT: Codex, GPT-x, etc are now widely used to generate code.

- How much are people ***using*** this generated code? Does it help?
- How ***good*** is this code?

Does Codex help coders In "Vivo"?

Understanding the Usability of AI Programming Assistants

Jenny T. Liang, Carnegie Mellon University, Pittsburgh, PA, USA, jtliang@cs.cmu.edu
Chenyang Yang, Carnegie Mellon University, Pittsburgh, PA, USA, cyang3@cs.cmu.edu
Brad A. Myers, Carnegie Mellon University, Pittsburgh, PA, USA, bam@cs.cmu.edu

ABSTRACT

The software engineering community recently has witnessed widespread deployment of AI programming assistants, such as GitHub Copilot. However, in practice, developers do not accept AI programming assistants' initial suggestions at a high frequency. This leaves

1. Usage Characteristics
- A Usage patterns
- B Motivation for using
- C Motivation for not using
- D Successful use cases

2. Usability of AI Programming Assistants

n=410, survey, Github Devs;
30% code generated;
Helps productivity
74% "<u>quick check</u>"
..but…Non-func reqmnts?
Hard to control?

Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models

Priyan Vaithilingam, pvaithilingam@g.harvard.edu, Harvard University, USA
Tianyi Zhang, tianyi@purdue.edu, Purdue University, USA
Elena L. Glassman, glassman@seas.harvard.edu, Harvard University, USA

ABSTRACT

Recent advances in Large Language Models (LLM) have made automatic code generation possible for real-world programming tasks in general-purpose programming languages such as Python. However, there are few human studies on the usability of these tools and how

on two different kinds of approaches: (1) program synthesis algorithms that search over a large program space defined by a domain-specific language (DSL) [2, 7, 10, 12, 14, 19, 24, 25, 30, 31, 34, 43], and (2) deep learning models that are trained on a large amount of existing code and can generate new code given some forms of

n=24; controlled study
+interview; @ Univ.
CoPilot not much help,
Many defects,
hard to Grok code,
..but subjects like it anyway!

Do LLMs help coders In “Vivo”?

BLOG ›

ML-Enhanced Code Completion Improves Developer Productivity

TUESDAY, JULY 26, 2022

Posted by Maxim Tabachnyk, Staff Software Engineer and Stoyan Nikolov, Senior Engineering Manager, Google Research

Update — 2022/09/06: *This post has been updated to remove a statement about an observed reduction in context switches that could not be confirmed with statistical significance.*

n=10k; Telemetry

3% code generated,

6% “Iteration” time reduction

>30% suggestion acceptance

Productivity Assessment of Neural Code Completion

Albert Ziegler
wunderalbert@github.com
GitHub, Inc.
San Francisco, USA

Eirini Kalliamvakou
ikaliam@github.com
GitHub, Inc.
San Francisco, USA

X. Alice Li
xalili@github.com
GitHub, Inc.
San Francisco, USA

Andrew Rice
acr31@github.com
GitHub, Inc.
San Francisco, USA

Devon Rifkin
drifkin@github.com
GitHub, Inc.
San Francisco, USA

Shawn Simister
narphorium@github.com
GitHub, Inc.
San Francisco, USA

Ganesh Sittampalam
hsenag@github.com
GitHub, Inc.
San Francisco, USA

Edward Aftandilian
eaftan@github.com
GitHub, Inc.
San Francisco, USA

n=2.6K; Survey + Telemetry

23%-28% suggestion acceptance

Acceptance rate correlates with self-reported productivity.

Personal take on Code LLMs

- Developers like them, Use them.
- Not clear they always fully understand the code they're using, and what the "PSP" is for this.
- *Prediction:* In an astonishingly short time, every computer: laptops, mobile phones, toasters, microwaves, air-traffic control, nuclear power plants, cruise missiles…

Will be running code generated by an LLM!!!

AI-generated code will
Be running
Everywhere!!

Do LLMs generate
buggy code?

Large Language Models and Simple, Stupid Bugs

Kevin Jesse
UC Davis
Davis, USA
krjesse@ucdavis.edu

Toufique Ahmed
UC Davis
Davis, USA
tfahmed@ucdavis.edu

Premkumar T. Devanbu
UC Davis
Davis, USA
ptdevanbu@ucdavis.edu

Emily Morgan
UC Davis
Davis, USA
eimorgan@ucdavis.edu

Methodology

- Simple, Stupid, Bugs 4 Java… One line bug fixes from 1000 projects. *(SStubs4J, Karampatsis & Sutton 2020)*
- Go back in history, and find when they were injected (by human dev)
- Try the with the prefix, and see…

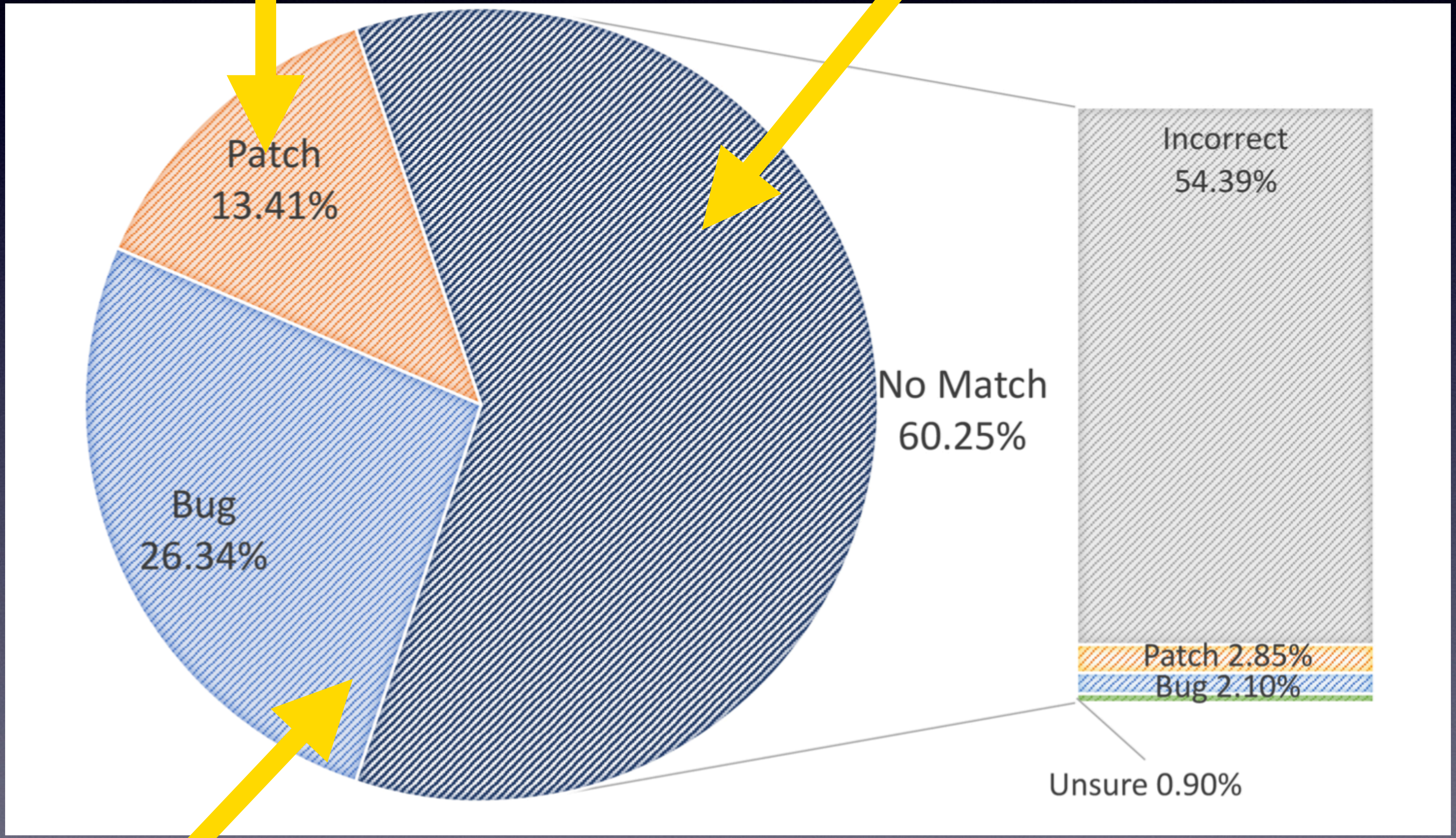
…….Does produce the 🪰 Or the 🩹

All samples in dataset used
were fixed before
LLM training data was gathered.

Result

Codex produces fixed code

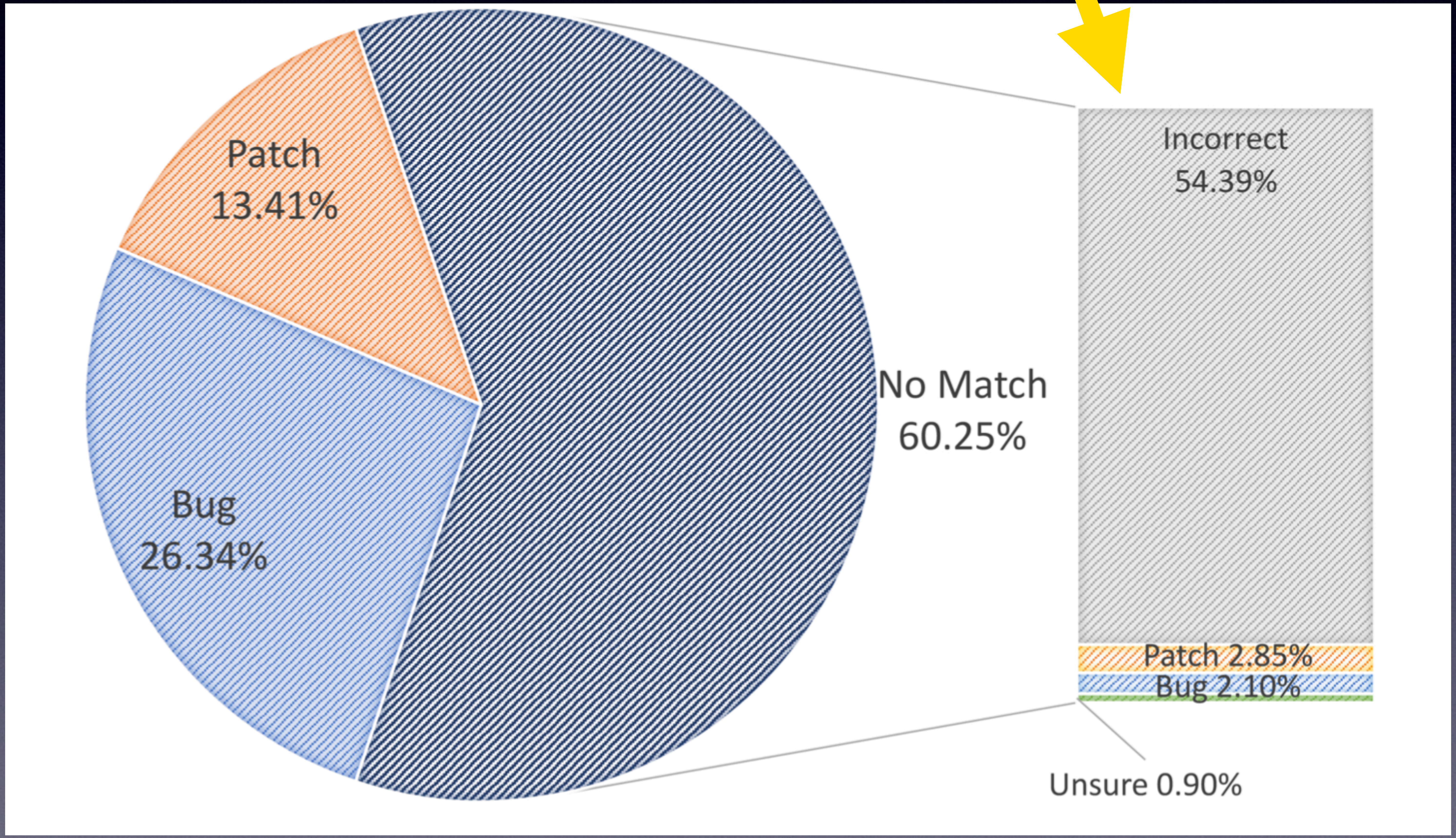
Something else



Codex produces buggy code TWICE as often

Result

Manual Review,
401 samples



Also looked at…

- When CoPilot generates Simple, Stupid Bugs, *were* they “stickier”?

- Good programmers Comment. Do Comments induce CoPilot not repeat human errors?

Take Aways…

- Programmers like LLM-based plugins.
- LLMs often recapitulate human errors.
- …when they do, these errors may be “sticky”.
- …but, we can improve their performance with comments.
- (Worry:) Devs use LLM-generated code without full review.