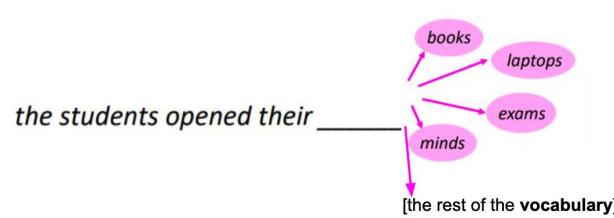


1. Introduction

Code Language Models

- In short, it is a language model **predicting words**

$$\prod_{i=1}^N p(x^{(i)} | \underbrace{x^{(1)}, \dots, x^{(i-1)}}_{\text{context}})$$

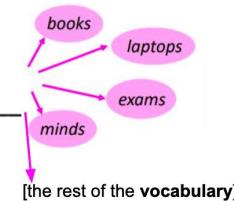


Code Language Models

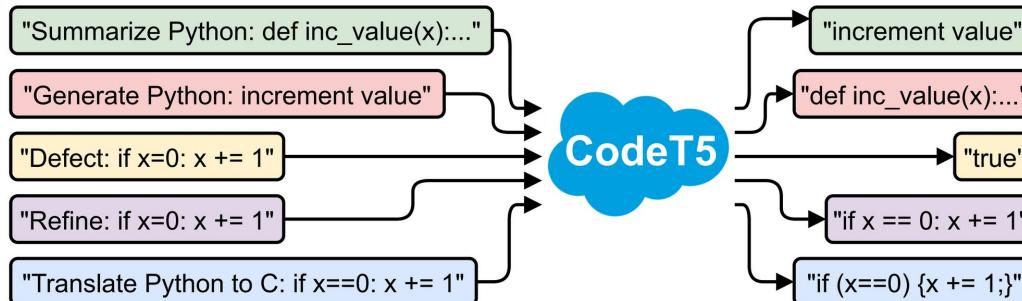
- In short, it is a language model **predicting words**

$$\prod_{i=1}^N p(x^{(i)} | \underbrace{x^{(1)}, \dots, x^{(i-1)}}_{\text{context}})$$

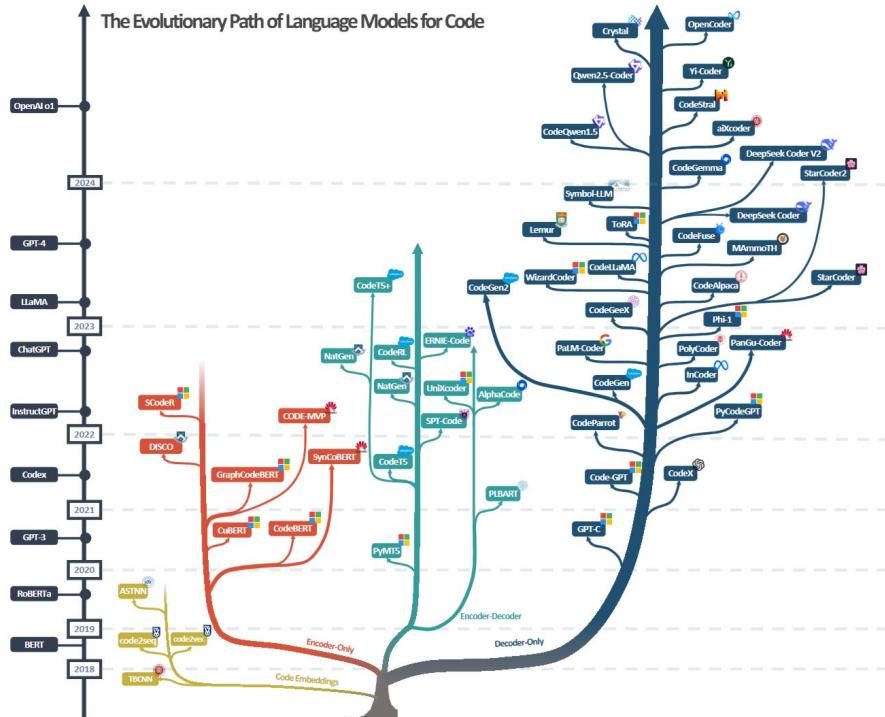
the students opened their _____



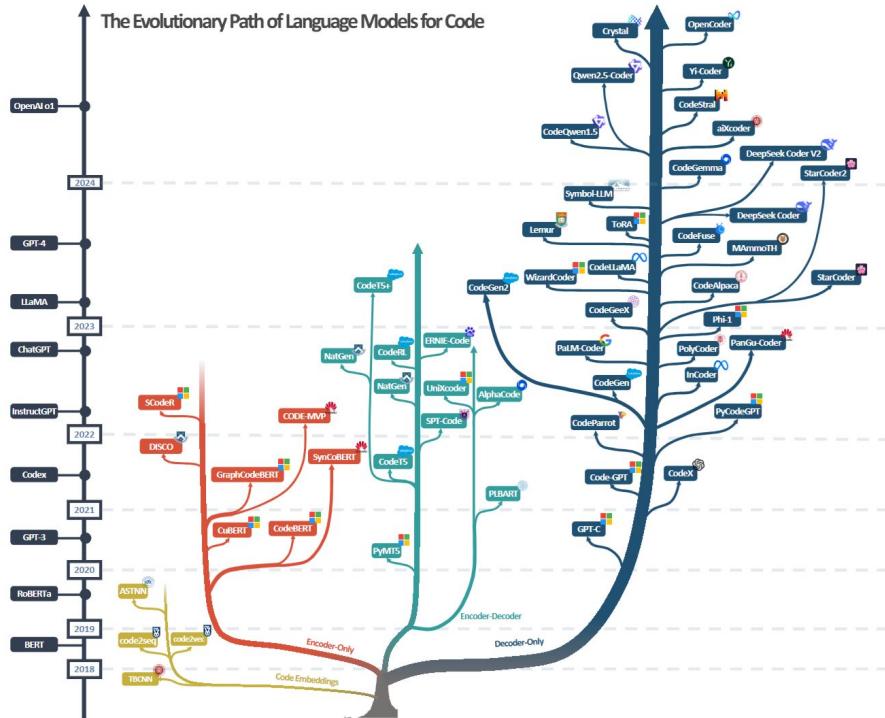
- It is mainly used to **predict code/programs**



The Age of Code Language Models

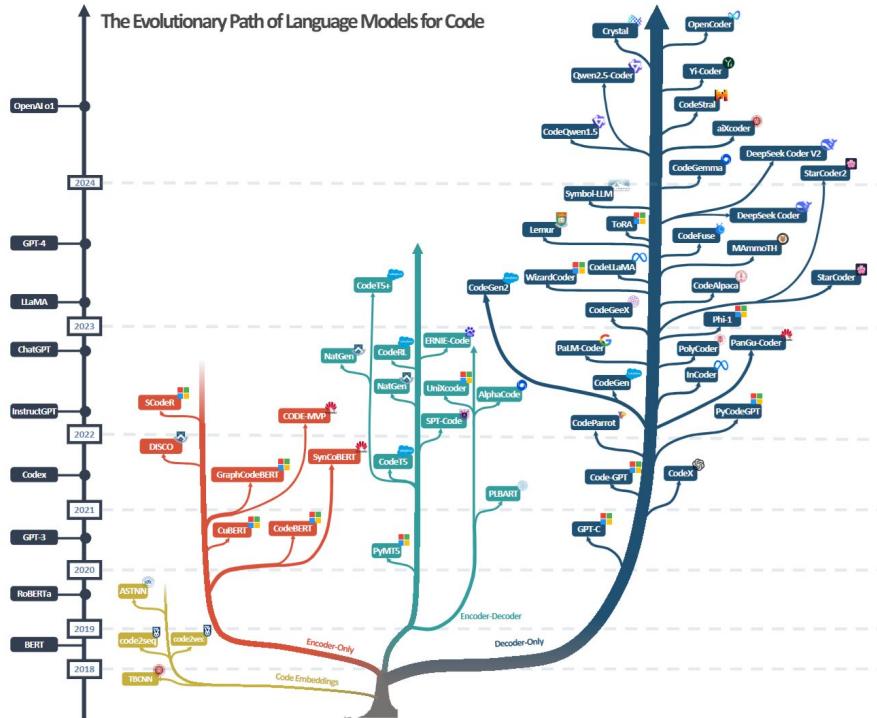


The Age of Code Language Models



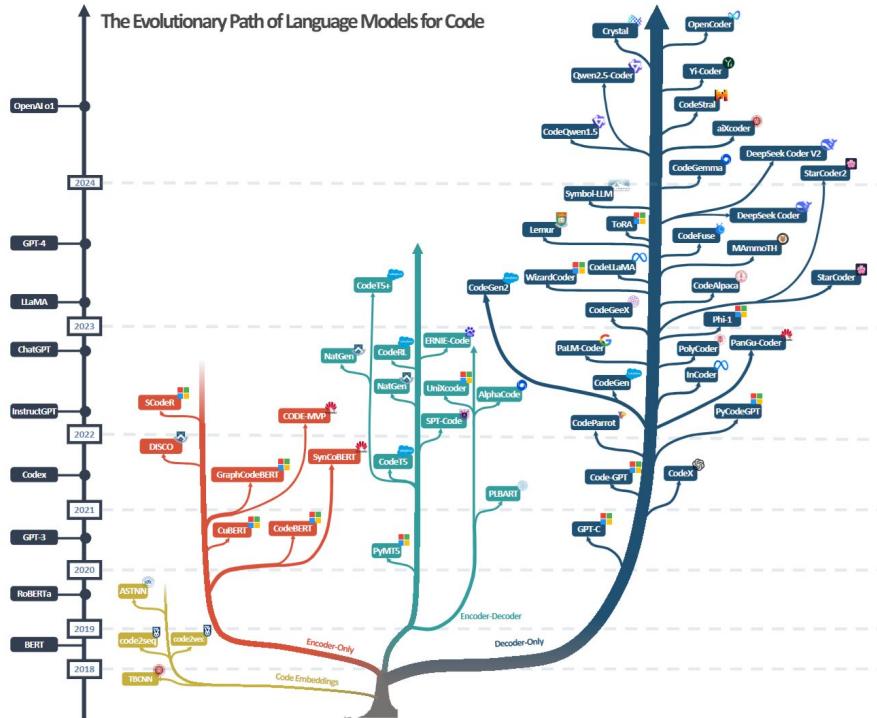
➤ (Mostly) Transformer-based

The Age of Code Language Models



- (Mostly) Transformer-based
- Decoder-Only Code LMs grow fast, mainly for code generation

The Age of Code Language Models



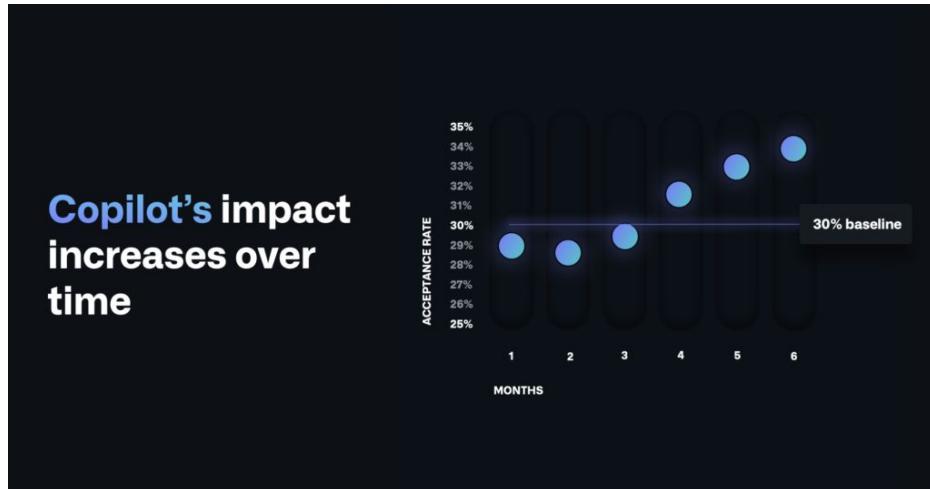
- (Mostly) Transformer-based
- Decoder-Only Code LMs grow fast, mainly for code generation
- From LMs trained on **code**, to LMs trained on **text and code**

Recent Trends of Code Language Models

- From Benchmarks to Real-world Applications

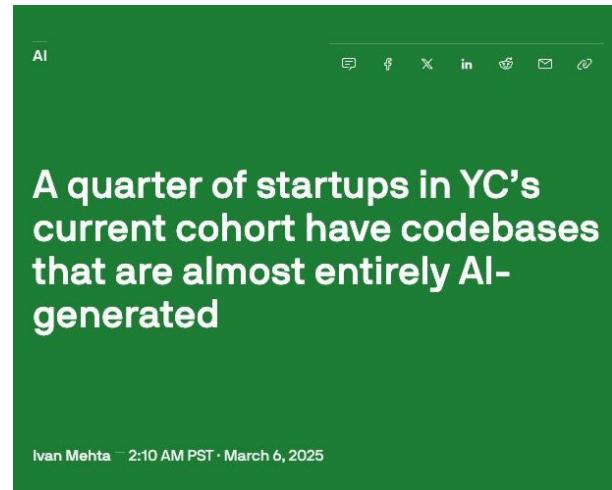
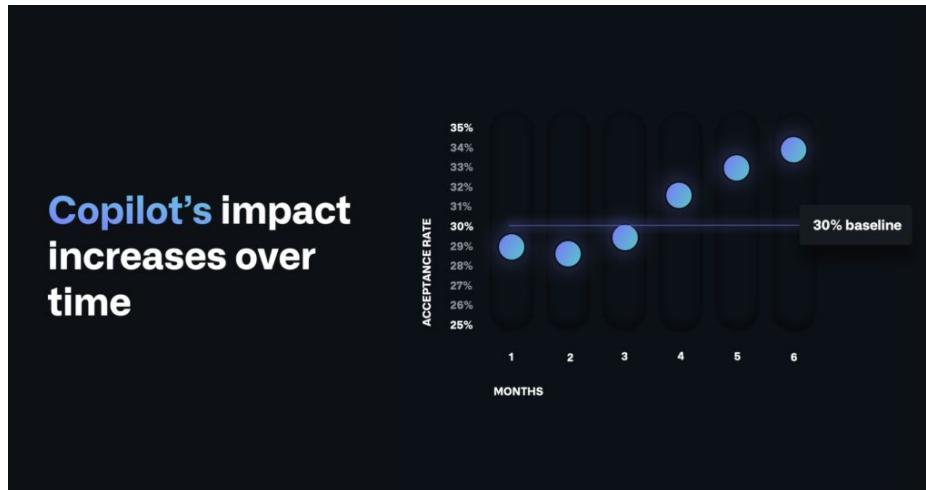
Recent Trends of Code Language Models

- From Benchmarks to Real-world Applications



Recent Trends of Code Language Models

- From Benchmarks to Real-world Applications



Recent Trends of Code Language Models

- Efficiency Matters!

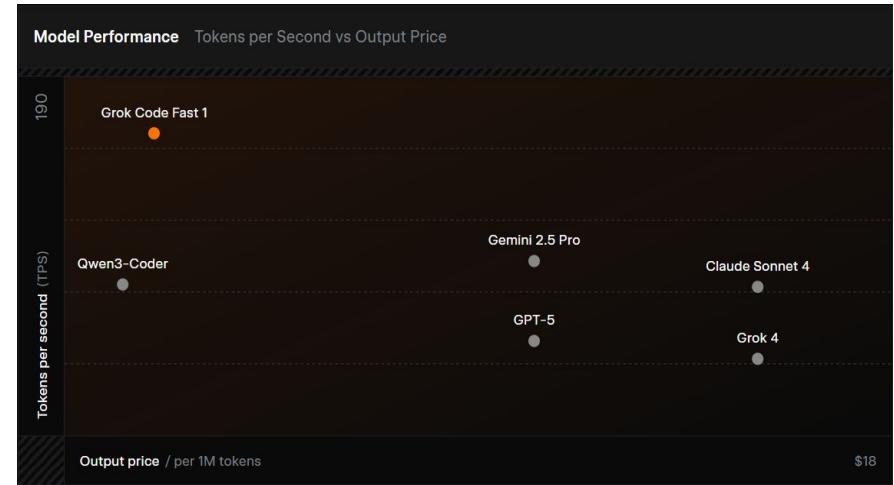
Recent Trends of Code Language Models

- Efficiency Matters!



Recent Trends of Code Language Models

- Efficiency Matters!



Before: Language Models for Code Generation

- Language models that **write code with you**

Before: Language Models for Code Generation

- Language models that **write code with you**

August 10, 2021 Product

OpenAI Codex

We've created an improved version of OpenAI Codex, our AI system that translates natural language to code, and we are releasing it through our API in private beta starting today.

[Start using Codex ↗](#)

Before: Language Models for Code Generation

- Language models that **write code with you**

August 10, 2021 Product

OpenAI Codex

We've created an improved version of OpenAI Codex, our AI system that translates natural language to code, and we are releasing it through our API in private beta starting today.

[Start using Codex ↗](#)

Introducing GitHub Copilot: your AI pair programmer

Today, we're launching a technical preview of GitHub Copilot, a new AI pair programmer that helps you write better code.

Technical preview

Your AI pair programmer

Now: Language Model Agents for Code

- Language model agents that **write code for you**

Now: Language Model Agents for Code

- Language model agents that **write code for you**

March 12, 2024

Introducing Devin, the first AI software engineer

By Scott Wu

Devin is a tireless, skilled teammate, equally ready to build alongside you or independently complete tasks for you to review.

With Devin, engineers can focus on more interesting problems and engineering teams can strive for more ambitious goals.

Now: Language Model Agents for Code

- Language model agents that **write code for you**

March 12, 2024

Introducing Devin, the first AI software engineer

By Scott Wu

Devin is a tireless, skilled teammate, equally ready to build alongside you or independently complete tasks for you to review.

With Devin, engineers can focus on more interesting problems and engineering teams can strive for more ambitious goals.

AI performance solving issues from 12 open-source Python repositories

SWE-bench Verified accuracy



CC-BY

epoch.ai

Goals of the Tutorial

- We will cover a number of key developments on Code Intelligence (2021 - 2025), with emphasis on the recent paradigms (2024 - 2025)

Goals of the Tutorial

- We will cover a number of key developments on Code Intelligence (2021 - 2025), with emphasis on the recent paradigms (2024 - 2025)
 - ❑ **Training**
 - ❑ **Evaluation**
 - ❑ **Applications and extensions**

Goals of the Tutorial

- We will cover a number of key developments on Code Intelligence (2021 - 2025), with emphasis on the recent paradigms (2024 - 2025)
 - ❑ Training
 - ❑ Evaluation
 - ❑ Applications and extensions
- This tutorial is **cutting-edge**, and there is still a long way to best develop and evaluate Code LMs for the real-world scenarios

Goals of the Tutorial

- We will cover a number of key developments on Code Intelligence (2021 - 2025), with emphasis on the recent paradigms (2024 - 2025)
 - ❑ **Training**
 - ❑ **Evaluation**
 - ❑ **Applications and extensions**
- This tutorial is **cutting-edge**, and there is still a long way to best develop and evaluate Code LMs for the real-world scenarios
 - ❑ **Existing research and key insights**
 - ❑ **Our perspectives on the current challenges & open problems**

Schedule

Time	Section	Presenter
9:00—9:15	Section 1: Introduction	Loubna/Terry
9:15—9:30	Section 2: Preliminaries	Terry
9:30—9:50	Section 3: Post-training Code LMs: Supervised Fine-Tuning	Wasi
9:50—10:15	Section 4: Post-training Code LMs: Reinforcement Learning	Binyuan
10:15—10:25	Q & A Session I	
 30min coffee break		
10:55—11:15	Section 5: Evaluating Code LMs: Function-level Code Generation	Terry
11:15—11:35	Section 6: Evaluating Code LMs: Repo-level & Agentic Code Generation	Zijian
11:35—11:55	Section 7: Bridging between Code and Natural Language	Qian
11:55—12:10	Section 8: Special Topics	Terry/Loubna
12:10—12:20	Section 9: Conclusion	Terry
12:20—12:30	Q & A Session II	