SCIENTIFIC
SOFTWARE
CENTER

UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

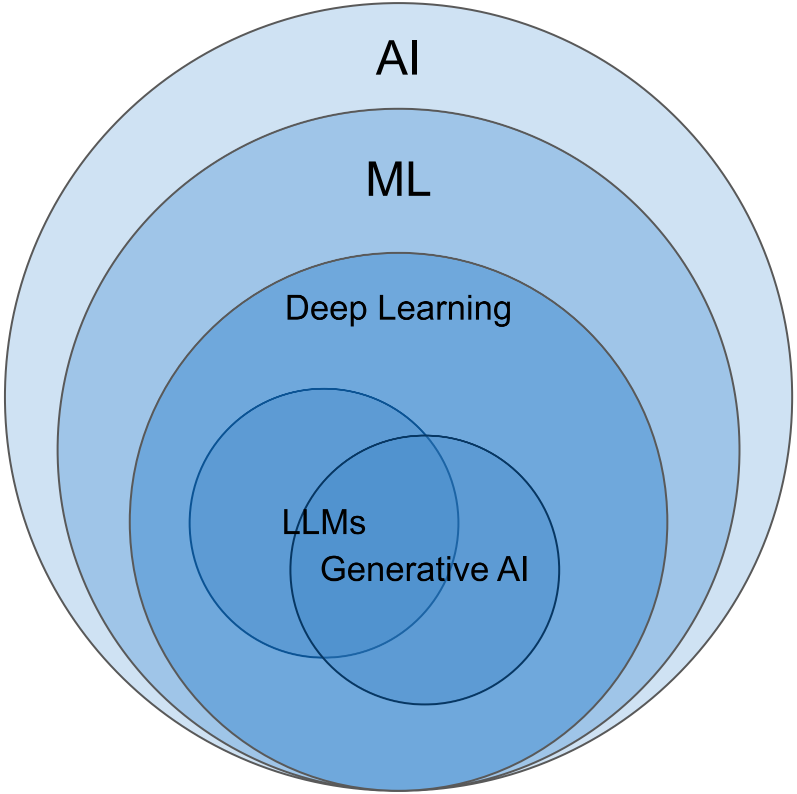
Generative AI for writing (research) software

Dr. Inga Ulusoy, Scientific Software Center

December 2024

Generative AI and GitHub Copilot

What is generative AI?



Discriminative model

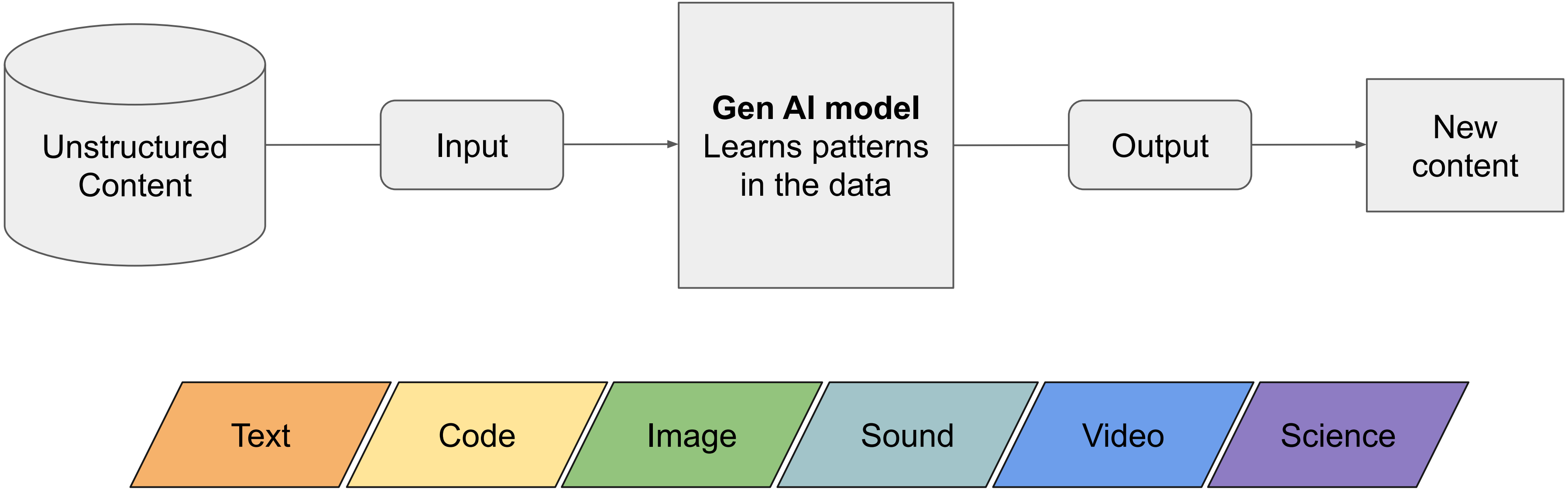
This is a cat.

Generative model

A cat is a small domesticated carnivorous mammal (Felis catus) that is commonly kept as a pet. Cats have been domesticated for thousands of years and have adapted to live alongside humans in various environments around the world. They are known for their agility, gracefulness, and independent nature. …

Photo by Krysten Merriman: https://www.pexels.com/photo/short-coated-gray-cat-20787/
Response by ChatGPT 3.5: https://chat.openai.com/share/846ff979-4eff-47a6-954c-dd2f2943fead

Generative AI Models



Unstructured Content

Input

Gen AI model
Learns patterns in the data

Output

New content

Text

Code

Image

Sound

Video

Science

Generative AI in academia

- ✔ Proposal writing
- ✔ Paper writing
- ✔ Visualization of research data
- ✔ Research software development
- ✔ For learning
- ✔ For teaching

! Openly state and reference use of GenAI

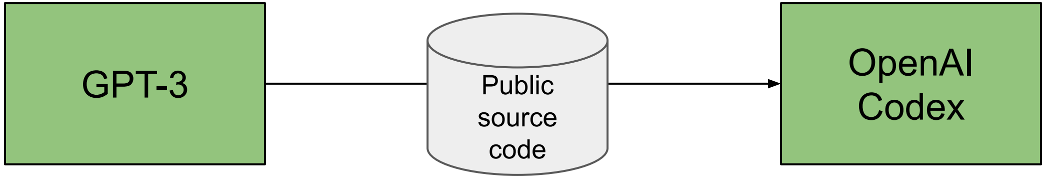
! Ensure no plagiarism occurs

! No infringement of intellectual property rights

X Writing reviews

https://www.dfg.de/resource/blob/289674/ff57cf46c5ca109cb18533b21fba49bd/230921-stellungnahme-praesidium-ki-ai-data.pdf

What is GitHub Copilot?



- Most capable in Python, but also JavaScript, Go, Perl, PHP, Ruby, Swift, TypeScript, Shell, …
- Generate code in response to natural language
- Good at mapping simple problems to libraries, APIs, functions
- Trained on 54 million public software repositories hosted on GitHub (May 2020), containing 179 GB of Python files, model has 12 billion parameters

https://doi.org/10.48550/arXiv.2107.03374

ChatGPT vs GitHub Copilot

ChatGPT

- Trained on a diverse dataset of text from the internet
- Takes in natural language prompt and returns human-like text response
- Static model
- Use through OpenAI chat

Copilot

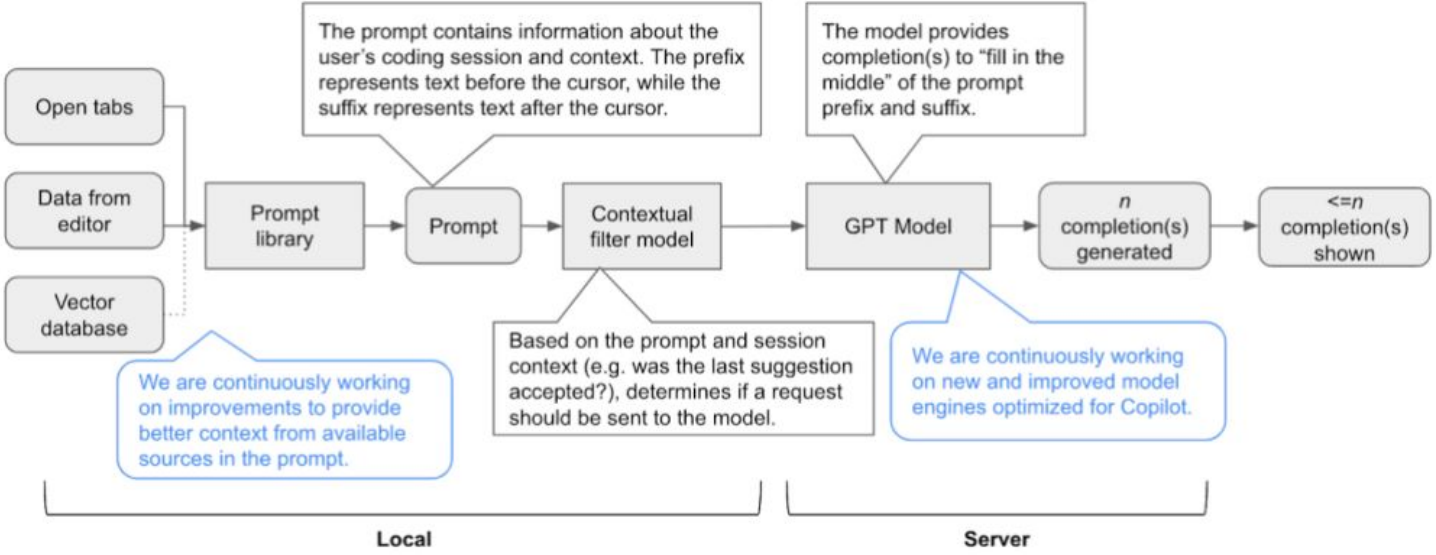
- Based on GPT-3
- Further trained on source code
- Takes in natural language and code prompt and returns code
- Static model, but is continuously updated
- Use through IDE, both in source code files and chat

Criticism

- Trained on open-source software, but is a paid service
- Could potentially leak personal data or produce something that looks like personal data
- Can suggest same code to different people, and the same code may also already exist in some repo with a specific open-source license (could have been part of the training): Check for duplicated code in Copilot settings
- Will replace developers in the long term, reducing them to code reviewers

Prompt engineering

- Generate clear instructions to guide AI systems
- “Single, Specific, Short, Surround”



The prompt contains information about the user’s coding session and context. The prefix represents text before the cursor, while the suffix represents text after the cursor.

The model provides completion(s) to “fill in the middle” of the prompt prefix and suffix.

Based on the prompt and session context (e.g. was the last suggestion accepted?), determines if a request should be sent to the model.

We are continuously working on improvements to provide better context from available sources in the prompt.

We are continuously working on new and improved model engines optimized for Copilot.

Simplified system diagram focused on model quality efforts. Made by Alice Li, machine learning researcher at GitHub.

https://github.blog/2023-05-17-how-github-copilot-is-getting-better-at-understanding-your-code/

How can generative AI assist you in writing software?

- Create content
- Provide language assistance
- Provide translation assistance (programming languages or different versions of the same programming language)
- Content summarization (explain code)
- Content curation (explain keywords/concepts)
- Writing assistance (style checking, best practices, identify errors and inconsistencies)
- Create tests and documentation

Using GitHub Copilot

How to set up GitHub Copilot

- You need a Copilot license - you should have set this up already per the instructions that I sent out
- Open VSCode, go to “Marketplace” and search for “Copilot”
- Install the “GitHub Copilot” extension, and the “GitHub Copilot Chat” extension
- Follow the prompts to sign in to GitHub (if you have not yet done so) and to set the permissions for Copilot
- You should be able to see Copilot code suggestions now
- You can configure Copilot on GitHub via your account settings
  - Enable or disable code duplication (allow or block code completion suggestions that match publicly available code)
  - Enable or disable prompt and suggestion collection

https://docs.github.com/en/copilot/using-github-copilot/getting-started-with-github-copilot
https://docs.github.com/en/copilot/configuring-github-copilot/configuring-github-copilot-settings-on-githubcom

How to use GitHub Copilot

- Start writing code to see suggestions
- Press `Tab` to accept suggestions, `Esc` to dismiss
- To see alternative suggestions, press Alt+] ([) (Windows/Linux), Option (⌥) or Alt+] ([) (macOS)

Example 1

Write a snippet of code that reads in a dataframe, and calculates statistics of the dataframe.

We will be using this dataset
https://www.kaggle.com/datasets/crawford/80-cereals

The starter repository is here

https://classroom.github.com/a/VqxSAOeR

Example 1

- Start typing code
- Start typing comments
- First write code, then the comments
- Write the comment and have copilot write the code for you
- You can also try this in an interactive environment (jupyter notebook) within VSCode, if you wish
- Ask the chat for specific advice

How to use GitHub Copilot

- Press Tab to accept suggestions, Esc to dismiss
- To see alternative suggestions, press Alt+] ([) (Windows/Linux), Option (⌥) or Alt+] ([) (macOS)
- Accept only the next word by pressing Control+→ (←) (Windows/Linux), Command+→ (←) (macOS)
- To accept next line, set custom shortcuts https://docs.github.com/en/copilot/configuring-github-copilot/configuring-github-copilot-in-your-environment#using-or-rebinding-keyboard-shortcuts-for-github-copilot
- Trigger inline suggestion Alt+\ (Windows/Linux), Option (⌥)+] (\) (macOS)
- Open GitHub Copilot Alt+Enter(Windows/Linux), Option (⌥) or Alt+Enter (macOS)
- See multiple options in a new tab Control+Enter

Example 2

Write a snippet of code either using numpy or the transformers library.

Starter repository: numpy

https://classroom.github.com/a/GXMIWM6m

Starter repository: transformers

https://classroom.github.com/a/el4hLsrr

Alternatively, you can try with another programming language.

What are your experiences?

Example 1

Example 2

Using GitHub Copilot: Best Practices

Good at:

- Following patterns already in the code-base using currently open tabs and context around cursor
- Python or other much-used programming language of public repos, like JavaScript
- Great first attempt at writing a function/method/snippet, but usually needs refactoring
- Helps you adhere to best practices and community coding standards
- Helps you write more readable and general code
- Speeds up code review
- Helps you maintain flow state through fewer context switches (stackoverflow/google search)
- Fosters test-driven development?
- Re-phrase requirements and look at them from different angle
- Creating boilerplate
- Commenting and summarizing/documenting code

Best practices

- Paste code into chat module and ask to have it explained
- Ask to explain code like a cooking recipe
- Ask for test suggestions including edge cases, negative testing, mocking
- Request suggestions for code style improvements
- Use meaningful names
- Involve code review from your coworkers (PRs)
- Be ready to explain your implementation and why you did it like that
- Don’t let it write long stretches of code, always little pieces to make sure to stay on the task

Day-to-day work

- You will write less code but review (read) more code and approve changes
- Copilot can write faster than you!
- Basically substitute for stackoverflow and google - less context switching?
- Good at things that are standard
- Can help you with libraries and interfaces
- Learns and adapts to your coding style (the more matching references in your context, the better the suggestions)
- Great at auto-completing comments
- Coding is not a continuous flow anymore, but a lot of reading
- Works better if you use meaningful names for variables, functions, …
- Frees you from writing obvious code and you are the architect, doing the real intellectual work (design and problem solving)
- You may learn new things from Copilots suggestions if you look them up and not just accept
- You will not save a lot of time as time spent doing actual coding is usually not that large of a portion compared to project planning and management - but your experience in coding will be different

What are the limits of generative AI for software writing?

- Lack of understanding: lack of context, coherence or relevance
- Varying quality and accuracy
- Limited creativity
- Ethical and bias concern
- Control and oversight
- Scalability and adaptability
- Limited context understanding
- Interpretability and explainability
- You still need to understand code to write code - and be good at it if you want to write good code

caveats

- Might spread bad code and lead to self-amplification (more public bad code, retraining on worse code samples, even worse code)
- Don’t fall into the trap of using code you do not understand! At least ask through the chat to have it explained to you, if you do not want to switch context
- If you are a very junior developer, it may take the joy of coding for you
- May replace junior developers!
- May render frameworks obsolete!
- May increase technical depth!
- Authorship and licensing complications, uncharted territory
- Understands context quite well, but not always the exact goal
- Cutting-edge or non-standard procedures do not receive great suggestions
- Creates dependencies and reduction of open knowledge

Privacy, Security, Licensing

Security (OWASP Top 10)

- GitHub Copilot itself is subject to typical security issues like prompt injection (see Gandalf), training data poisoning, sensitive information disclosure
- Insecure output handling (accept output without scrutiny) - you may introduce vulnerabilities in your code if you do not understand the suggestion fully
- Overreliance (overly relying on the tool without general oversight) - you may introduce misinformation, legal issues or security vulnerabilities
- GitHub Copilot is subject to attackers, possibly causing denial of service

Privacy

- User Engagement Data: accepted or dismissed completions, error messages, system logs, and product usage metrics (stored for two years)
- Prompts: inputs for chat or code along with context (not retained)
- Suggestions: AI-generated code lines or chat responses provided to users (not retained)
- Feedback Data: real-time user feedback, including reactions (e.g., thumbs up/down) and optional comments, along with feedback from support tickets ("stored for as long as needed for its intended purpose")
- Data from individual licenses (ie, student licenses) is used for training (not the case for GitHub Enterprise or Business) - same as with ChatGPT

Licensing

- If you incorporate third-party code, you need to make sure there are no license incompatibilities
- However, with Copilot you may use another person's code unaware
- How will you provide attribution/copyright?

GitHub is being sued because of this https://hackernoon.com/doe-vs-github-ammended-complaints-on-copyright-infridgement-open-source-licenses-and-more

You may block suggestions that match public code, but this only works from 150+ characters (ignoring whitespace).

GitHub announced in 2022 that they will introduce a feature providing a reference to matching public code https://github.blog/2022-11-01-preview-referencing-public-code-in-github-copilot/

Summary

Summary: About GitHub Copilot

- Based on OpenAI Codex
- Takes in code or text
- Generates code and comment blocks / docstrings
- Works in your IDE and thus reduces context switching
- Commercial tool based on open-source repositories
- May create code that matches public code which has been attributed to certain license terms
- Depending on your subscription plan, your code snippets may be used for retraining the model

Summary: Using GitHub Copilot

- Get suggestions
- Toggle suggestions
- Open tab with multiple options
- Use the chat for more general questions/answers
- Limited to certain IDE
- Limited to type of Copilot subscription

Summary: When and when not to use GitHub Copilot

- Summarizing code
- Writing tests
- Commenting code
- Boilerplate code
- Much-used APIs
- Writing assistance

- Special cases
- Special libraries/programming languages
- Understand your code
- Stay on the task
- Make sure to adhere to ethical, legal and security constraints

Summary: Security, Privacy and Licensing

- Consider: Privacy of the data and of the source code
- Consider: Security risks (due to the LLM or your trust in it)
- Consider: License attribution

If you have more strict requirements on the above three, either do not use Copilot or use it with the proper settings/appropriate plan!