

Automatizzare la documentazione di codice legacy con IA Generativa



Speaker:

Giovanni Rosa, PhD

AI Engineer (R&D)

@Technology Reply

 giovannirosa.com

 gio.rosa@reply.it



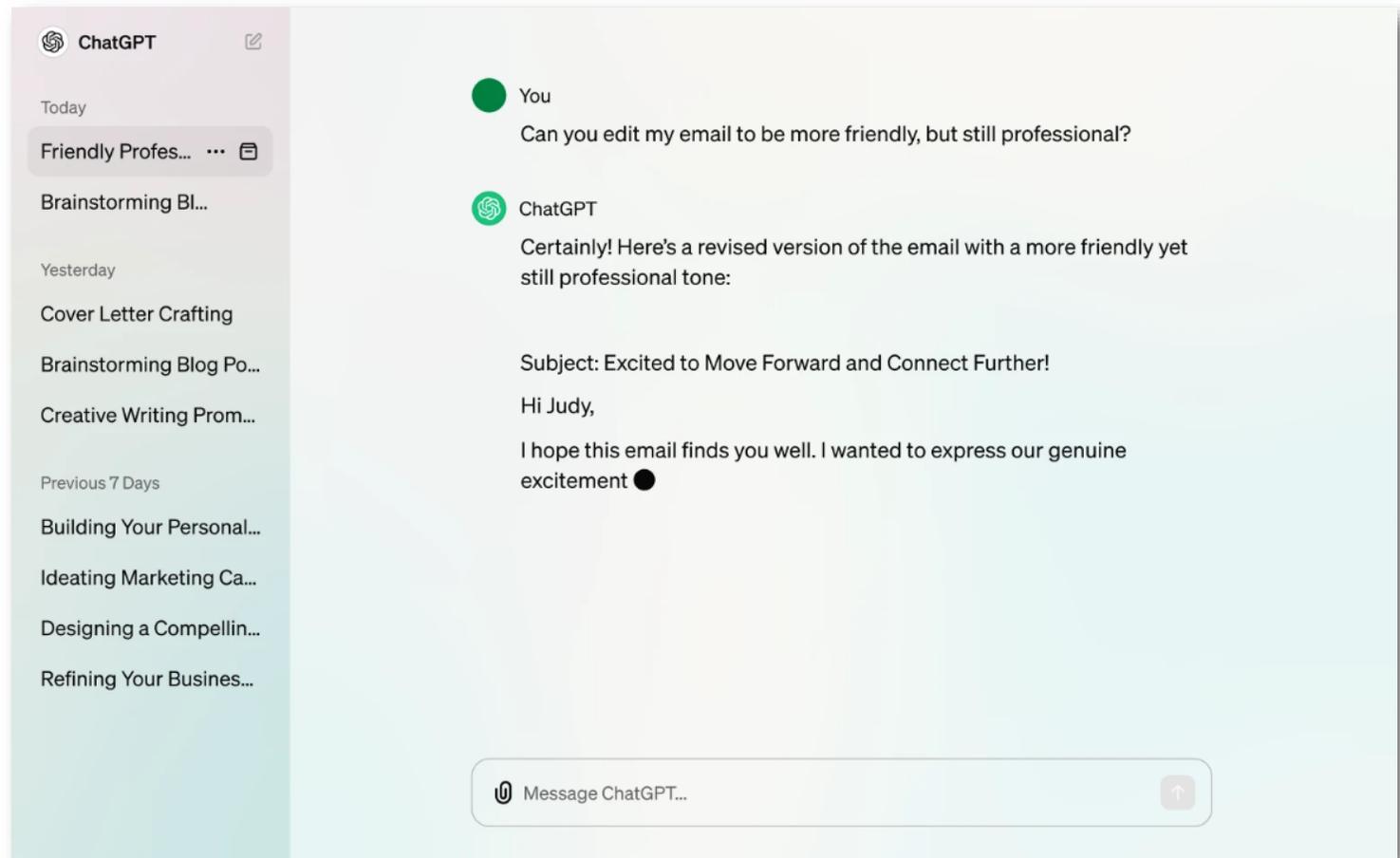
**November
2022**



ChatGPT

Chat Assistant

powered by GenAI



THE DAILY AI NEWS

www.dailyainews.com

FOR THE LATEST FICTIONAL AI WORLD NEWS

Dec 2022

Is this the Death of Google Search and the rise of ChatGPT?

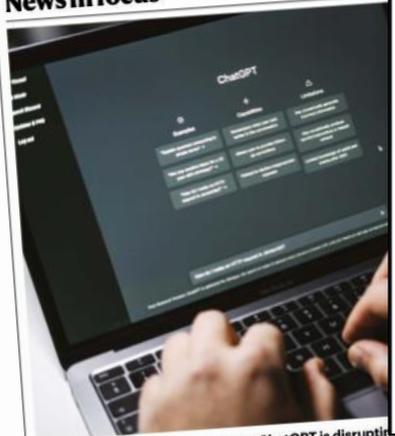


How would you describe the front page of a newspaper that reads "Google search is dead, long live ChatGPT!"

"Why is ChatGPT going to replace google search?"

It is not specified why ChatGPT is

News in focus



The artificial-intelligence chatbot ChatGPT is disrupting

CHATGPT LISTED AS AUTHOR ON RESEARCH PAPERS

Many scientists disapprove of articles crediting the AI tool as a co-author.

By Chris Stokel-Walker

The artificial-intelligence (AI) chatbot ChatGPT that has taken the world by

papers. But some publishers say that an AI's contribution to writing papers can be acknowledged in sections other than the author list. (Nature's news team is editorially independent.)

HealthyNation Tech

Opinion How ChatGPT will transform medicine this year

Though still in its initial phase, the platform is already cutting down the time needed to conduct medical scientific research

BY FAUSTINE NGILA

That is just one among billions of medical answers for many health questions stored inside this artificial brain. Its accuracy has long been discussed



ChatGPT: Optimizing Language Models for Dialogue

With ChatGPT, health administrators can now announce an official end to unnecessary scanning of paperwork. PHOTO: FILE

solutions, treatments and consultations. And with Kenya's tech ecosystem booming, where over 130 health tech startups have sprouted over the last seven years.

tools. "We are not new to these tools. He says that *Oncoscience* peer reviewed this paper after he asked its editor to do so. The journal did not respond to *Nature's* request for comment.

A fourth article⁴, co-written by an earlier chatbot called GPT-3 and posted on French preprint server HAL in June 2022, will soon be published in a peer-reviewed journal, says co-author Almira Osmanovic Thunström, a neurobiologist at Sahlgrenska University Hospital in Gothenburg, Sweden. She says one journal rejected the paper after review, but a second accepted it with GPT-3 as an author after she rewrote the article in response to reviewer requests.

THE DAILY AI NEWS

www.dailyainews.com

FOR THE LATEST FICTIONAL AI WORLD NEWS

Dec 2022

Is this the Death of Google Search and the rise of ChatGPT?



How would you describe the front page of a newspaper that reads "Google search is dead, long live ChatGPT!"

"Why is ChatGPT going to replace google search?"

It is not specified why ChatGPT is



ChatGPT vs software developers: is generative AI the end of the road for developers?

News in focus



The artificial-intelligence chatbot ChatGPT is disrupting

HealthyNation Tech

Opinion How ChatGPT will transform medicine this year

Though still in its initial phase, the platform is already cutting down the time needed to conduct medical scientific research

BY FAUSTINE NGILA

Technologists have long discussed



ChatGPT: Optimizing Language Models for Dialogue

We've trained a model called ChatGPT, which is designed to be conversational. It can follow instructions to generate text in a variety of formats, such as articles, stories, and code. It can also be used to generate text that is more specific to a user's needs. For example, it can generate text that is tailored to a specific audience or topic. This means it can improve efficiency in the pharmaceutical supply chain by automating the process of generating purchase orders, invoices, and delivery notes.

The Freeman Lifestyle

Jobs that Could be Replaced by AI

EDITOR: YASUNARI RAMON SUAREZ TAGUCHI SATURDAY | February 18, 2023

Since its rollout in November last year, the artificial intelligence matrix developed by the OpenAI group known as ChatGPT has been used to do all sorts of things – from writing cover letters to coming up with well-written essays.

Essentially a chatbot, many have come to see it as more than a program that's designed to automate queries in customer service chats – with Google even alleged to have said that it could "hire" ChatGPT as an entry level coder if it applied for a position at the company without the knowledge that it was a program and not a real person.

With this development, the question could replace have been brought up. Here are taken on five professions which industry experts and tech pundits are saying could be replaced by AI.



Graphic Designers



Customer Service Agents

Though the prospect of "talking" to a machine, as opposed to a human customer service agent, is not appealing to most, the customer service field is largely at risk due to advances in the field of AI. Chances are, you have already

CHATGPT LISTED AS AUTHOR ON RESEARCH PAPERS

Many scientists disapprove of articles crediting the AI tool as a co-author.

By Chris Stokel-Walker

The artificial-intelligence (AI) chatbot ChatGPT that has taken the world by

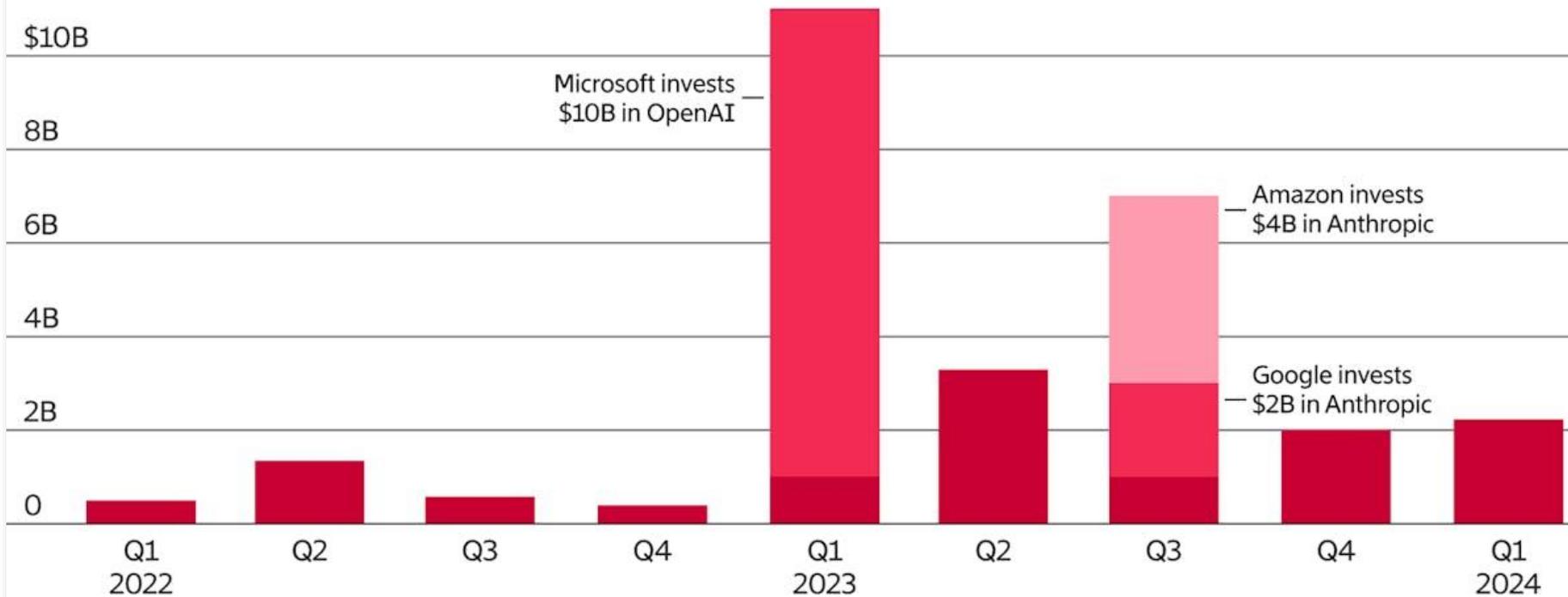
papers. But some publishers say that an AI's contribution to writing papers can be acknowledged in sections other than the author list. (Nature's news team is editorially independent)

tools. "We are not new to this," he says. He says that *Oncoscience* peer reviewed this paper after he asked its editor to do so. The journal did not respond to *Nature's* request for comment.

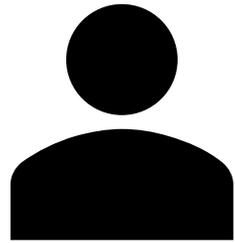
A fourth article, co-written by an earlier chatbot called GPT-3 and posted on French preprint server HAL in June 2022, will soon be published in a peer-reviewed journal, says co-author Almira Osmanovic Thunström, a neurobiologist at Sahlgrenska University Hospital in Gothenburg, Sweden. She says one journal rejected the paper after review, but a second accepted it with GPT-3 as an author after she rewrote the article in response to reviewer requests.

Generating Investment

The release of ChatGPT in November 2022 sparked a rush of investment in generative artificial intelligence companies.



Source: The Information reporting



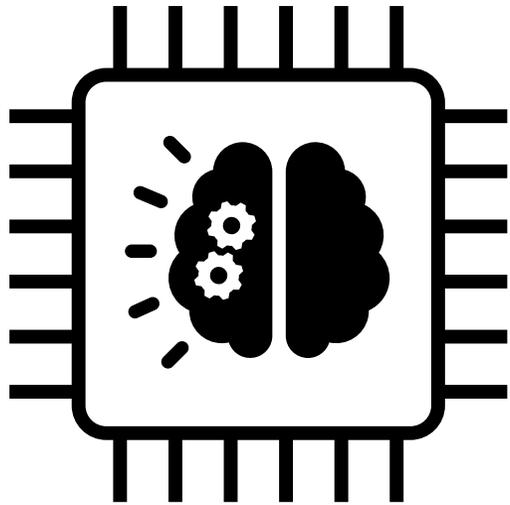
User

call



1

“Generate an image logo for the Italian Oracle User Group 2024 event hosted by Oracle in Milan city”



Generative AI
model

create



2



By DALL-E 3

Oracle AI Apps (Finance, Human Resources, Sales, Service, Procurement)

AI services



OCI Generative AI



Digital Assistant



Speech



Language



Vision



Document
Understanding



Anomaly
Detection

Machine Learning Services

OCI Data Science



ML in Oracle Database



Data Labeling



Data

AI Infrastructure

Compute Bare Metal
Instances and VMs



Cluster Networking



Storage and
HPC File systems

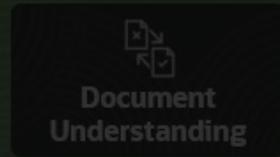
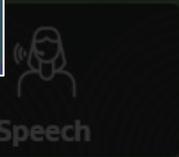
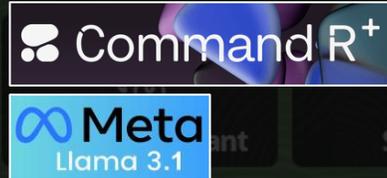


Source: blogs.oracle.com

Oracle AI Services

Oracle AI Apps
(Finance, Human Resources, Sales, Service, Procurement)

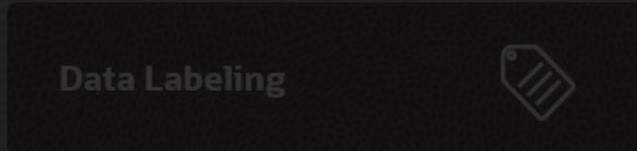
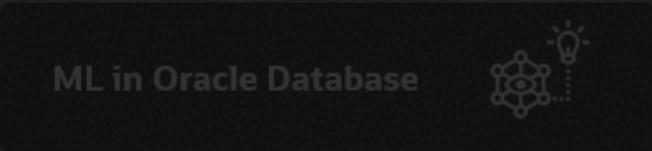
😊 Inference API (pay-per-use)



😊 Dedicated instances

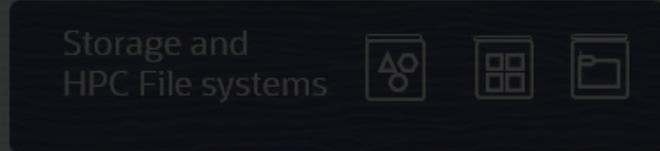
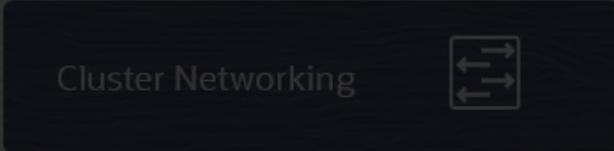
😊 Fine-tune models

Machine Learning Services



Data

AI Infrastructure



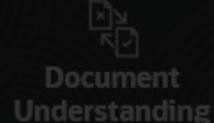
Source: blogs.oracle.com

Oracle AI Services

Oracle AI Apps

(Finance, Human Resources, Sales, Service, Procurement)

😊 Inference API (pay-per-use)



😊 Dedicated instances

😊 Fine-tune models

Machine Learning Services

ML in Oracle Database



Data Labeling



😊 Local SLM - no GPU!

😊 Vector store

😊 AutoML

Compute Bare Metal Instances and VMs



Cluster Networking



Storage HPC

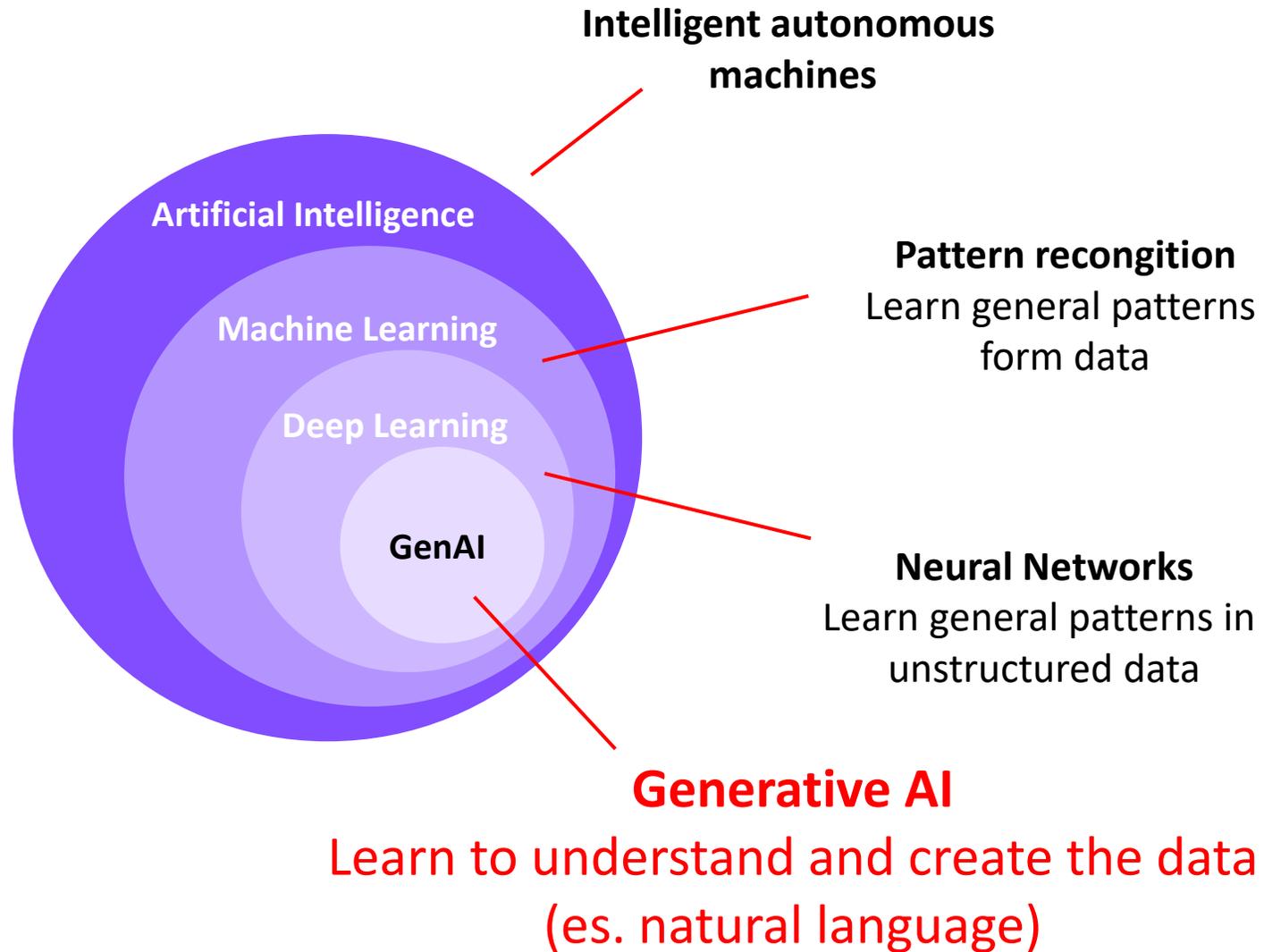


Source: blogs.oracle.com

Oracle AI Services

What is
Generative AI



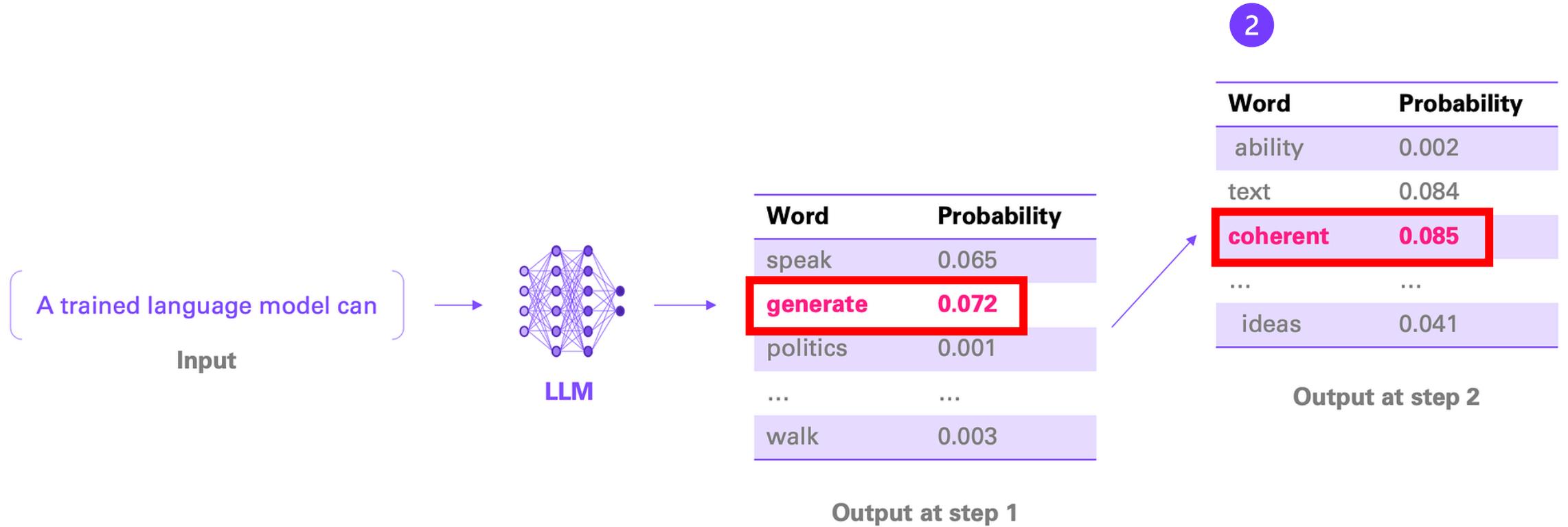


Large
Language
Model



LLM

A trained language model can generate coherent...



A trained language model can generate...

1

Source: medium.com

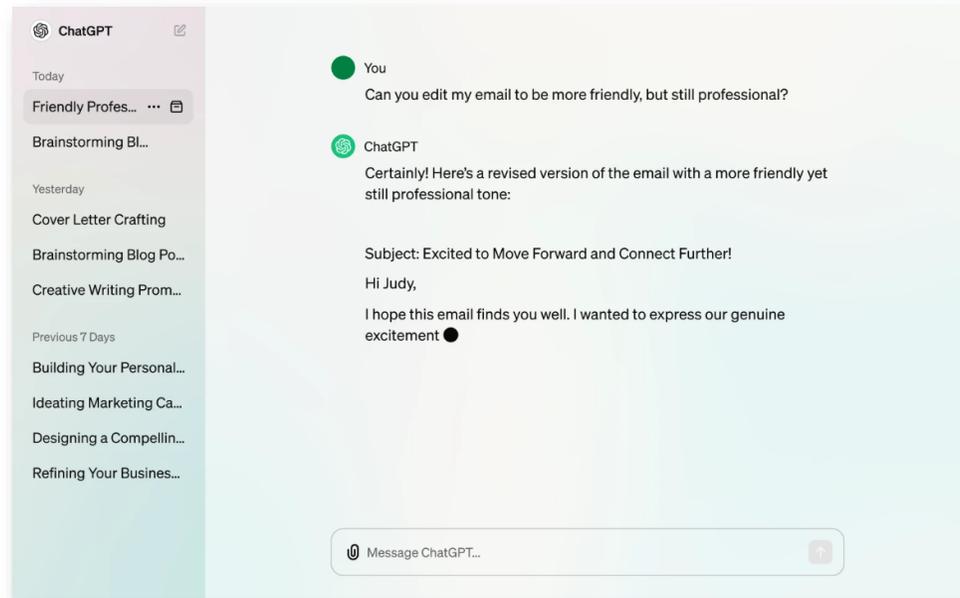
How does a language model work?

G

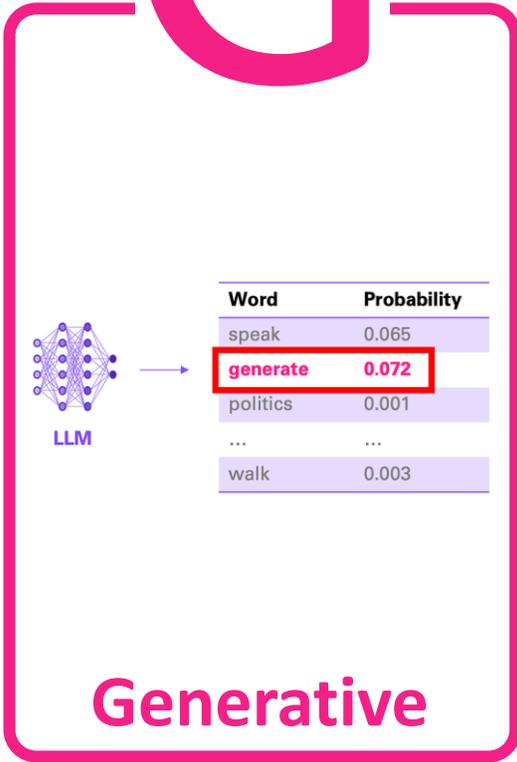
P

T

3.5



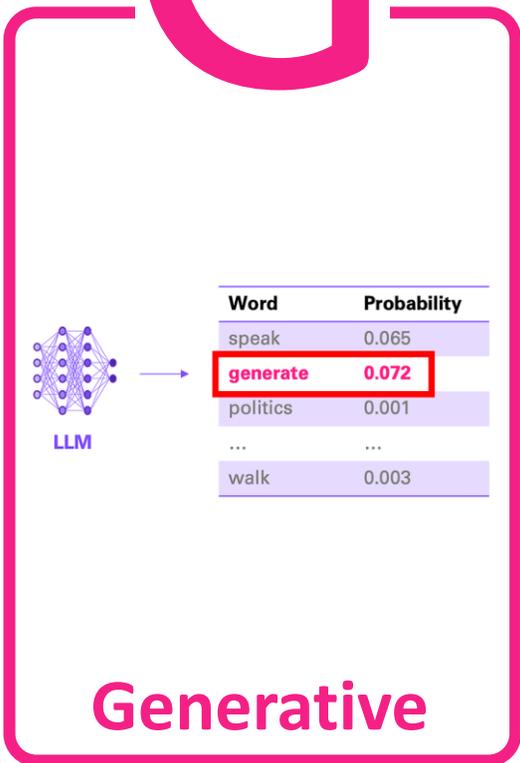
G



P

T

G



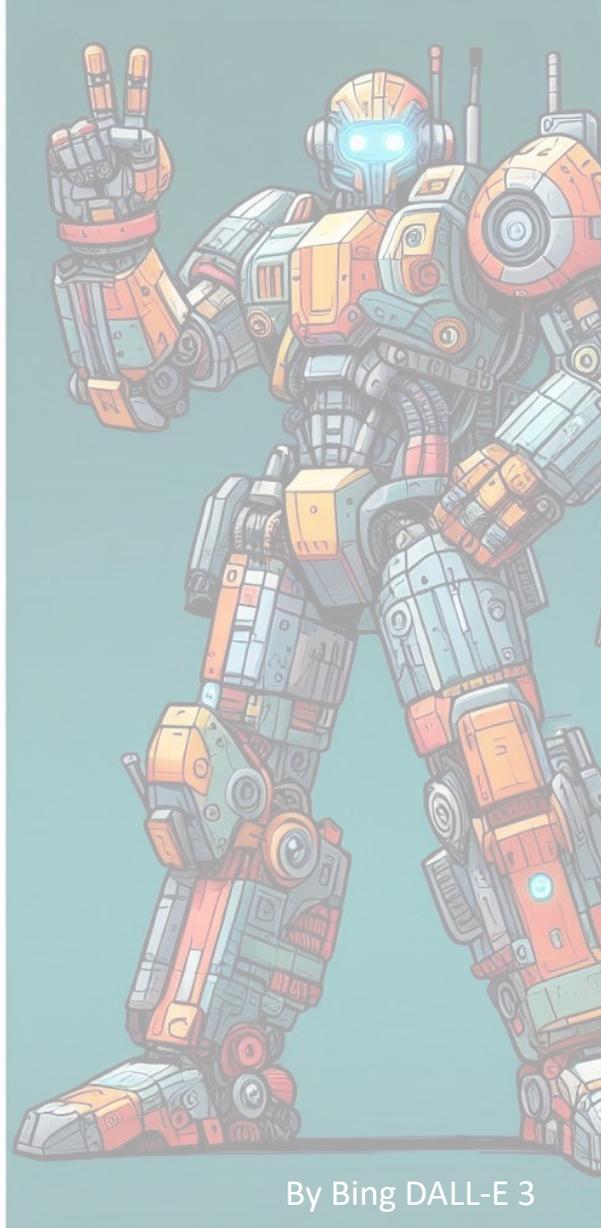
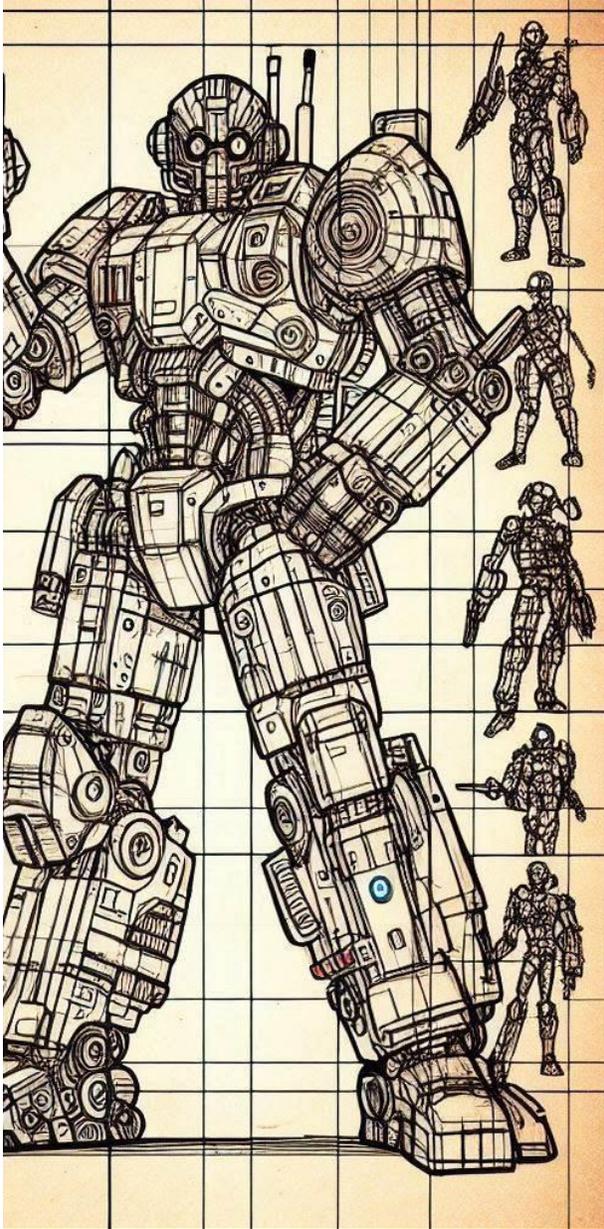
P



T

Pre-training

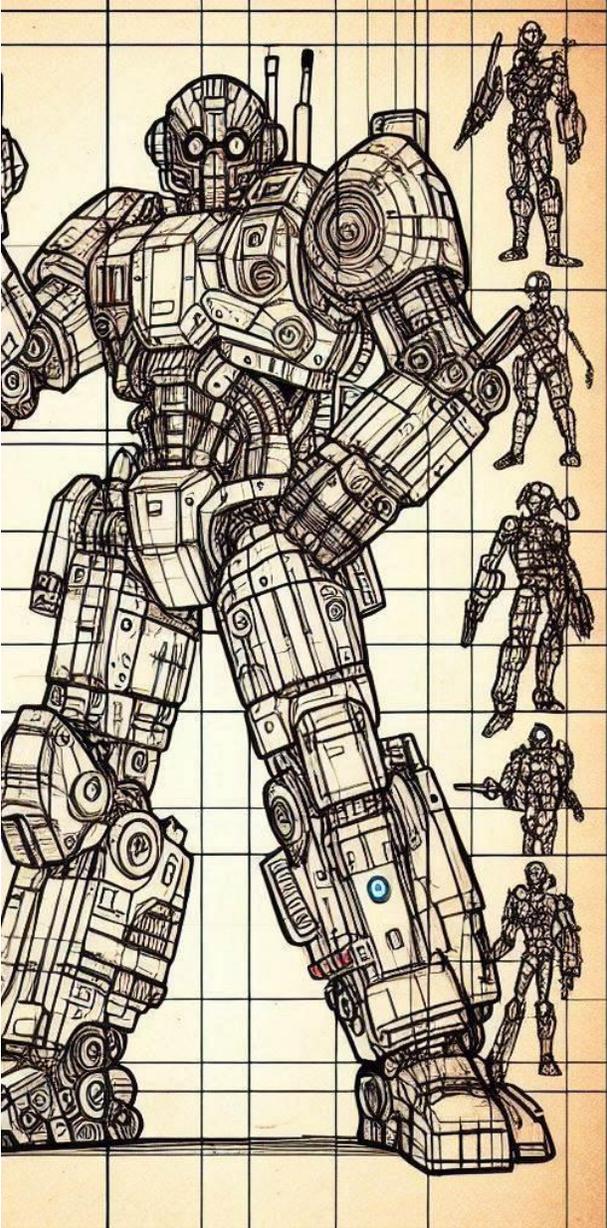
Learn
language structure
and knowledge



How to train a «Large» Language Model?

Pre-training

Learn
language structure
and knowledge



Instruction Fine-tuning

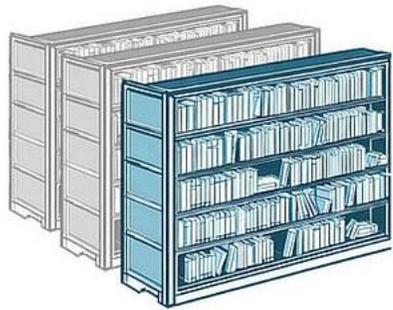
Learn
how to chat



How to train a «Large» Language Model?

GPT-4

Est. train size



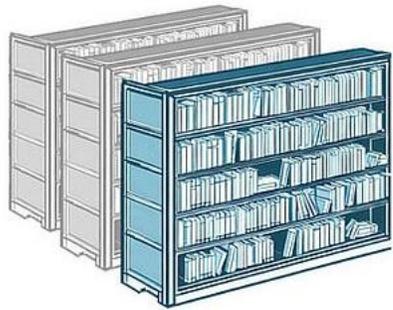
650 km

Line of library shelves

Pre-training Large Language Models

GPT-4

Est. train size



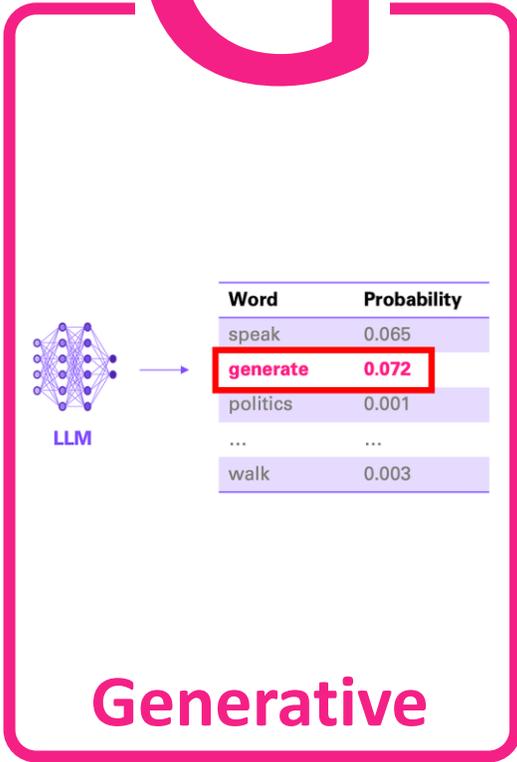
650 km

Line of library shelves



Pre-training Large Language Models

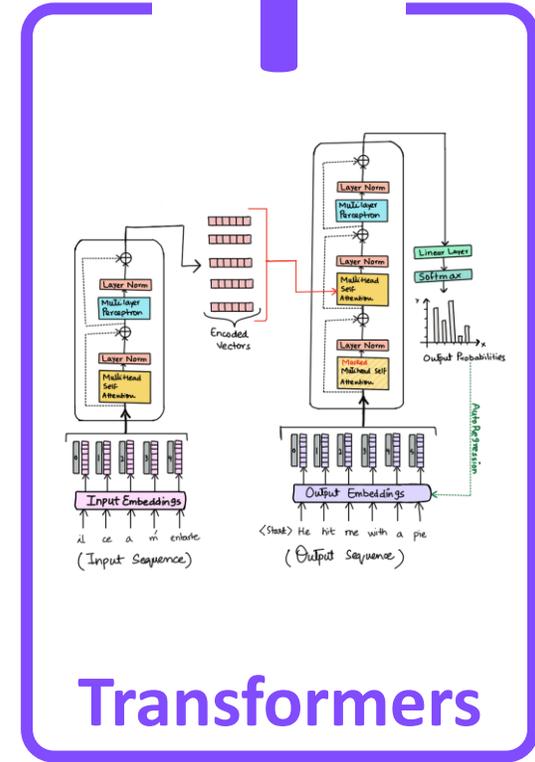
G

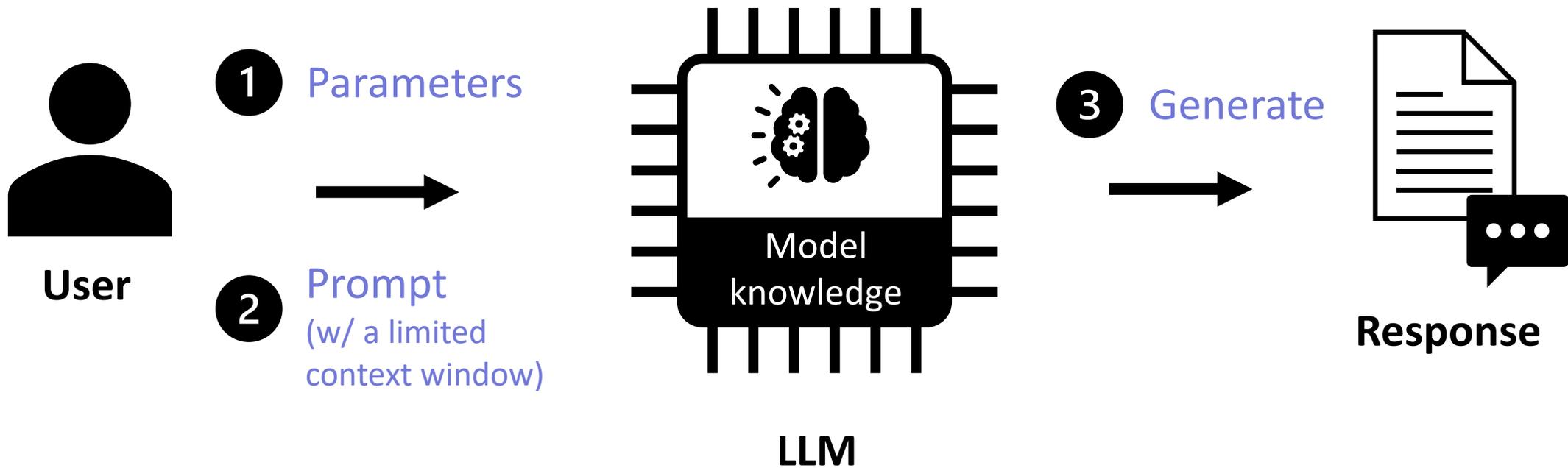


P

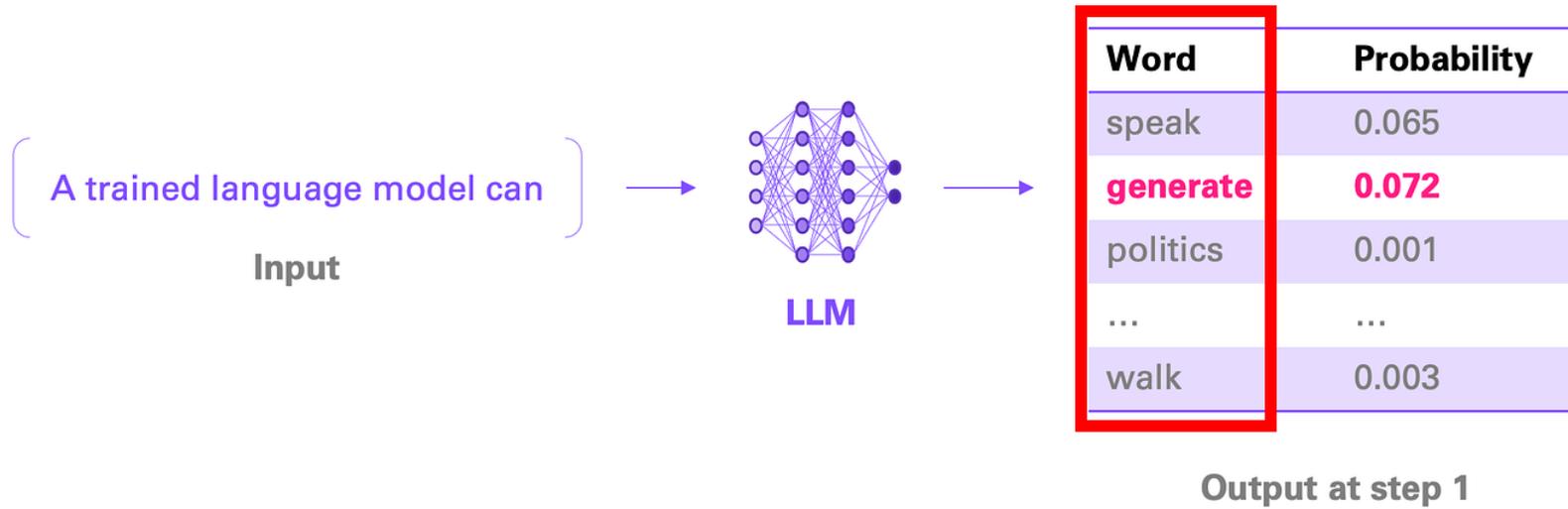


T



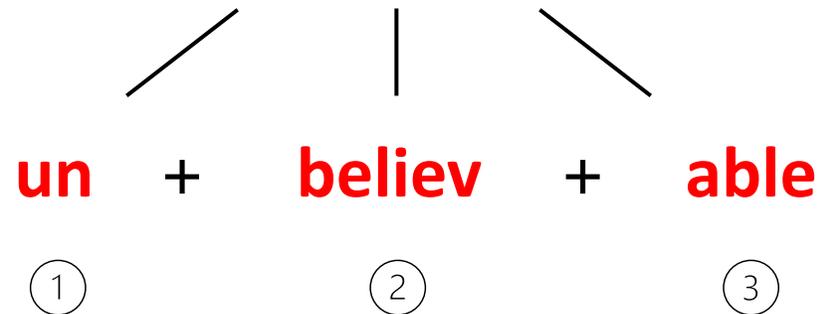


How to instruct LLMs ?



... [generate] [**unbelievable**] [textual] [sentences]

1 word ≈ 2-3 tokens



Source: medium.com

How does a language model work?

Generative AI

Overview

Playground

Chat

Embedding

Dedicated AI clusters

Custom models

Endpoints

Scope

Compartment

Chat ?

To get started, choose a model and a preset prompt example. Then, refine the prompts and parameters to fit your use cases. See [model types](#) for more information. All model responses have [moderation filtering](#) applied for explicit content. Note that some models have deprecation/retirement dates. View our [model list](#) for more details.

Model

cohere.command-r-plus v1.2

View model details

Example

Choose example

View code

Hi! Who are you?



I am an AI-assistant chatbot trained to assist human users by providing thorough responses. I am always learning and improving based on user feedback. How can I help you today?

Prompt

2

Parameters

1

Parameters

Maximum output tokens ⓘ

 600

Preamble override ⓘ

Provide concise responses

Temperature ⓘ

 0.25

Top p ⓘ

 0.75

Top k ⓘ

 0

Frequency penalty ⓘ

 0

Presence penalty ⓘ

 0

Type a message...

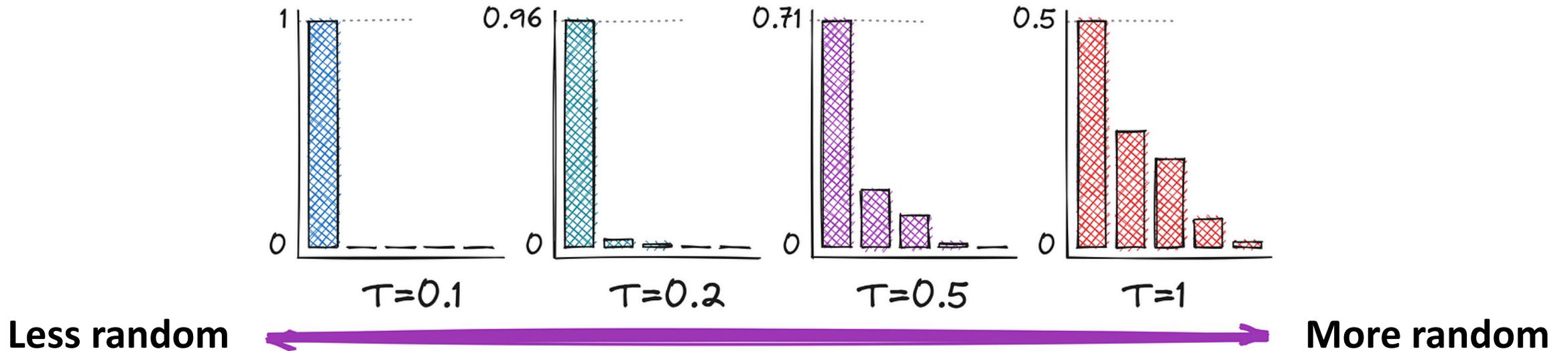
Submit

Clear chat

OCI Generative AI Playground

Temperature

controls randomness and creativity



Source: pub.towardsai.net

Model: cohere.command-r-plus v1.2 | View model details | Example: Choose example | View code

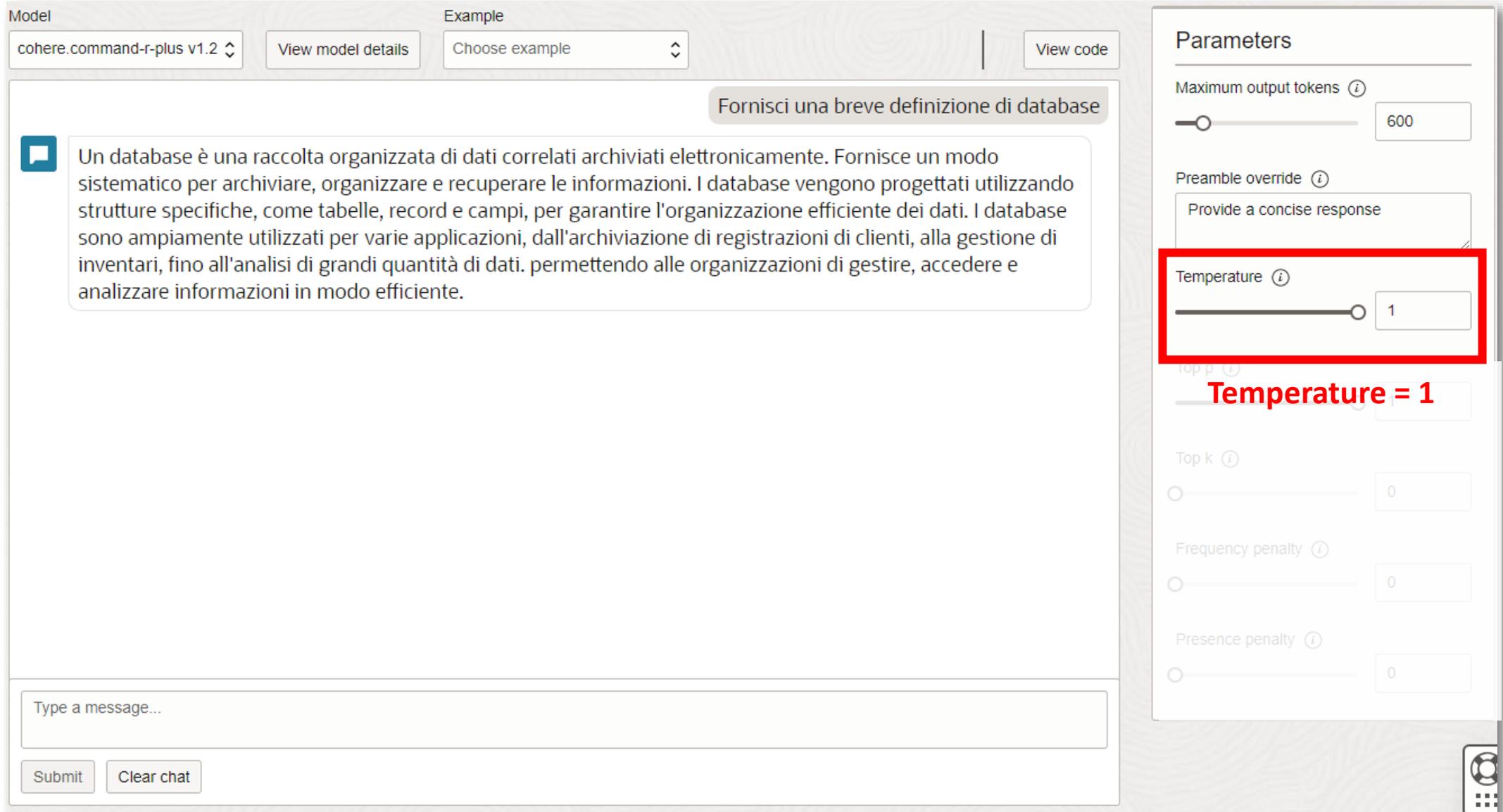
Fornisci una breve definizione di database

Un database è una raccolta organizzata di dati correlati archiviati elettronicamente. Fornisce un modo sistematico per archiviare, organizzare e recuperare le informazioni. I database vengono progettati utilizzando strutture specifiche, come tabelle, record e campi, per garantire l'organizzazione efficiente dei dati. I database sono ampiamente utilizzati per varie applicazioni, dall'archiviazione di registrazioni di clienti, alla gestione di inventari, fino all'analisi di grandi quantità di dati, permettendo alle organizzazioni di gestire, accedere e analizzare informazioni in modo efficiente.

Type a message... | Submit | Clear chat

Parameters

- Maximum output tokens: 600
- Preamble override: Provide a concise response
- Temperature: 1**
- Top p: 0
- Top k: 0
- Frequency penalty: 0
- Presence penalty: 0



1

Temperature

Model: cohere.command-r-plus v1.2 | View model details | Example: Choose example | View code

Fornisci una breve definizione di database

Un database è una raccolta organizzata di dati strutturati, archiviati e accessibili elettronicamente in un sistema informatico. I database vengono utilizzati per archiviare, gestire e recuperare in modo efficiente grandi quantità di informazioni.

Type a message... | Submit | Clear chat

Parameters

- Maximum output tokens: 600
- Preamble override: Provide a concise response
- Temperature: 0**
- Top p: 0
- Top k: 0
- Frequency penalty: 0
- Presence penalty: 0

1

Temperature

Prompting example:
User review classification

Step 0: System Prompt



SYSTEM

You are an advanced NLP classification model.

Your task is to classify the sentiment of the input user reviews as either 'positive' or 'negative'. The reviews are referring to products of an e-commerce platform.

Return as a response ***ONLY ONE*** of the following strings:

- "Positive", for positive sentiment;
- "Negative", for negative sentiment.

Step 1: User Instructions

SYSTEM

You are an advanced NLP classification model.

Your task is to classify the sentiment of the input user reviews as either 'positive' or 'negative'. The reviews are referring to products of an e-commerce platform.

Return as a response ***ONLY ONE*** of the following strings:

- "Positive", for positive sentiment;
- "Negative", for negative sentiment.

USER

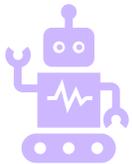
Il servizio clienti non funziona. Ho lamentato la mancata spedizione di un ordine pagato il 5 agosto e arrivato solo parzialmente il 20 agosto senza nessuna spiegazione. Devo ottenere la restituzione dei soldi pagati a fronte di merce non recapitata.



Step 2: Model Response

USER

Il servizio clienti non funziona. Ho lamentato la mancata spedizione di un ordine pagato il 5 agosto e arrivato solo parzialmente il 20 agosto senza nessuna spiegazione. Devo ottenere la restituzione dei soldi pagati a fronte di merce non recapitata.



ASSISTANT

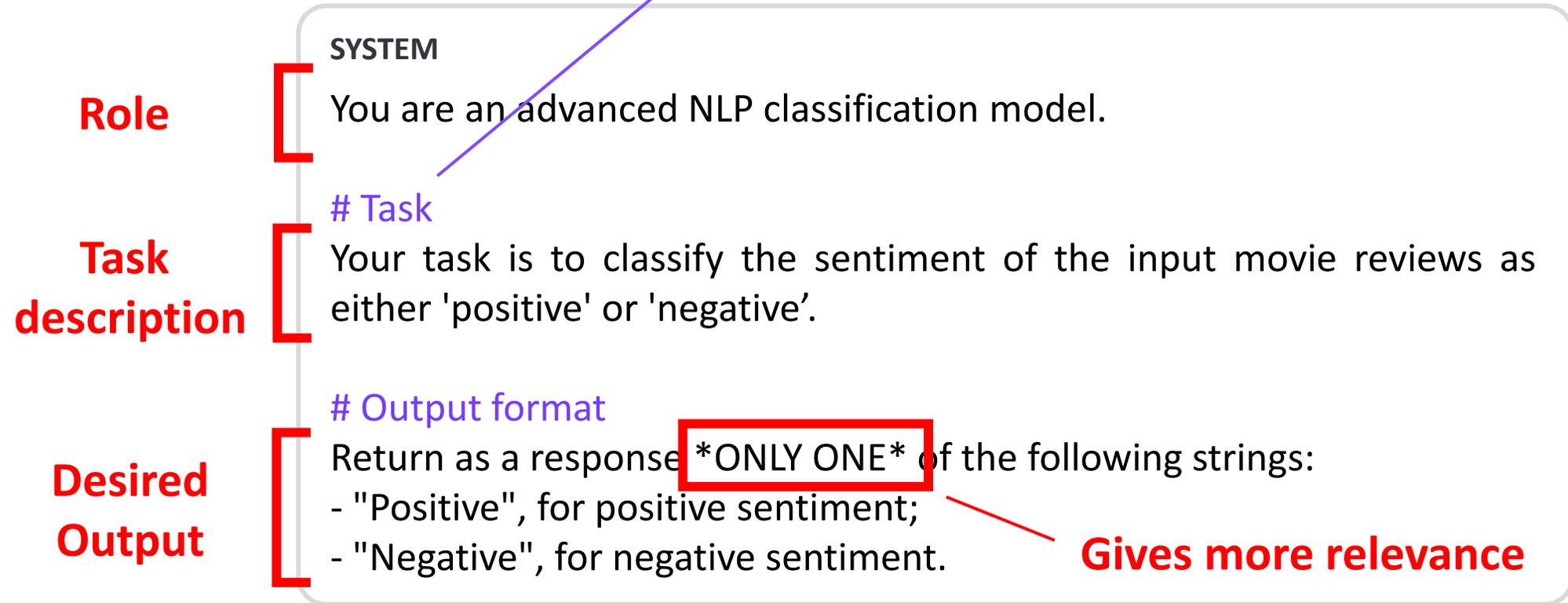
Negative

Prompt Engineering



More at: www.promptinguide.ai

Section delimiters



How to write a good prompt?

Beware of Hallucinations!



- ✗ Ambiguous prompt
- ✗ Anomalies in training data
- ✗ Outdated model knowledge

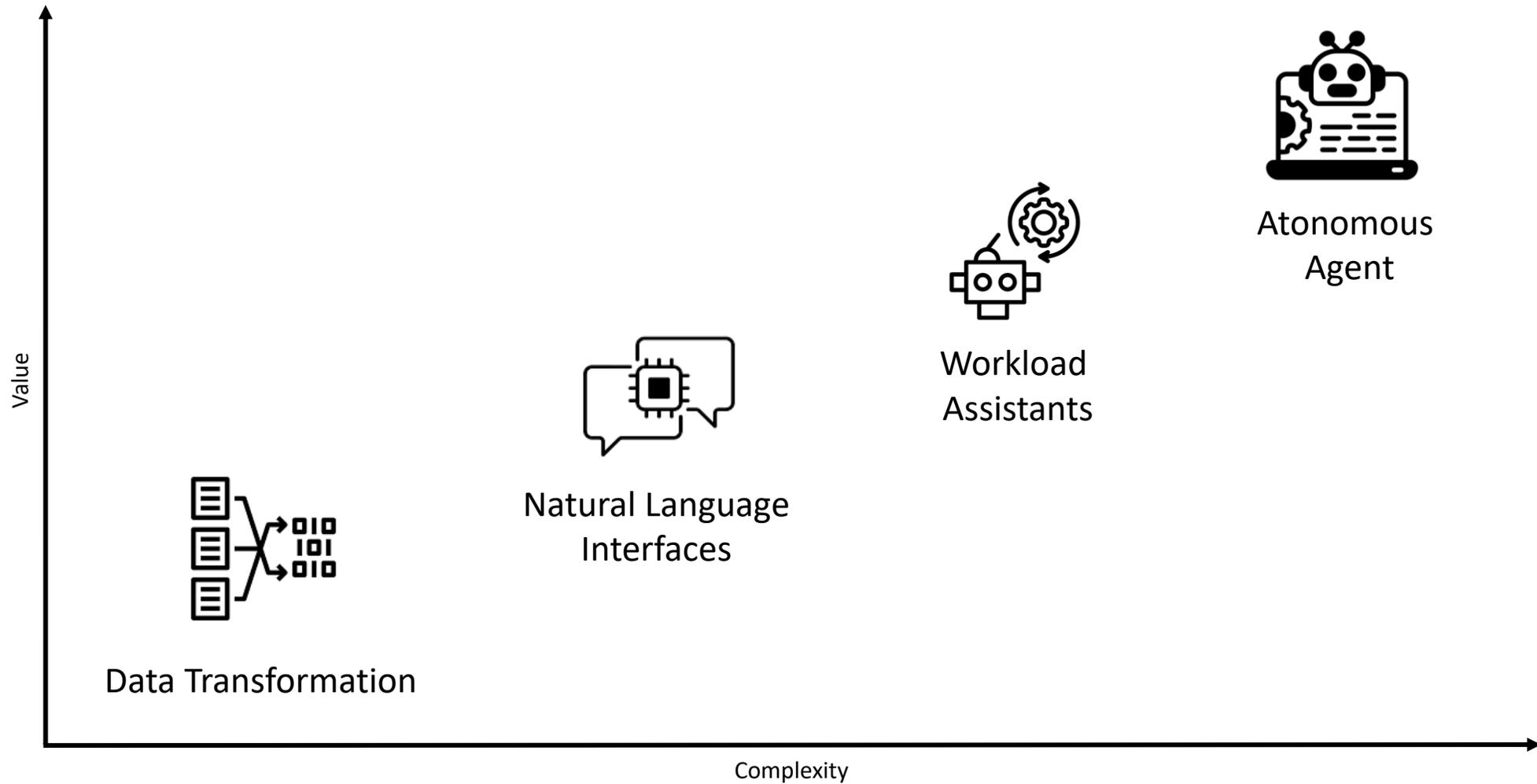


What's the capital of Mars?

The capital of Mars is Muskland.



GenAI Use Cases



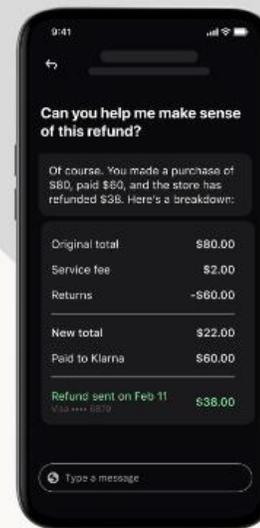
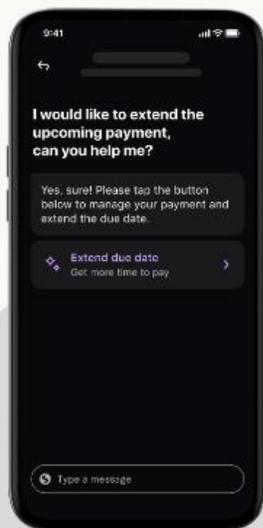
GenAI use cases

Klarna AI assistant handles two-thirds of customer service chats in its first month

February 27, 2024



Klarna
Klarna



Klarna

```
JS test.js 1 ●
JS test.js > calculateDaysBetweenDates
1 function calculateDaysBetweenDates(begin, end) {
    var beginDate = new Date(begin);
    var endDate = new Date(end);
    var days = Math.round((endDate - beginDate) / (1000 * 60 * 60 * 24));
    return days;
}
2
```



How much does AI impact development speed? An enterprise-based randomized controlled trial

Elise Paradis, Kate Grey, Quinn Madison, Daye Nam,
Andrew Macvean, Nan Zhang, Ben Ferrari-Church, Satish Chandra

Paradis et al. 2024

arXiv:2410.12944v1 [cs.SE] 16 Oct 2024

productivity? To date, the software engineering literature has provided a range of answers, targeting a diversity of outcomes: from perceived productivity to speed on task and developer throughput. Our randomized controlled trial with 96 full-time Google software engineers contributes to this literature by sharing an estimate of the impact of three AI features on the time developers spent on a complex, enterprise-grade task. We found that AI significantly shortened the time developers spent on task. Our best estimate of the size of this effect, controlling for factors known to influence developer time on task, stands at about 21%, although our confidence interval is large. We also found an interesting effect whereby developers who spend more hours on code-related activities per day were faster with AI and future research considerations are discussed. In we invite further research that explores the impact of AI on ecosystem level and across multiple suites of AI-enhanced tools since we cannot assume that the effect size obtained in this study will necessarily apply more broadly, or that the AI found using internal Google tooling in the summer of 2024 will translate across tools and over time.

I. INTRODUCTION

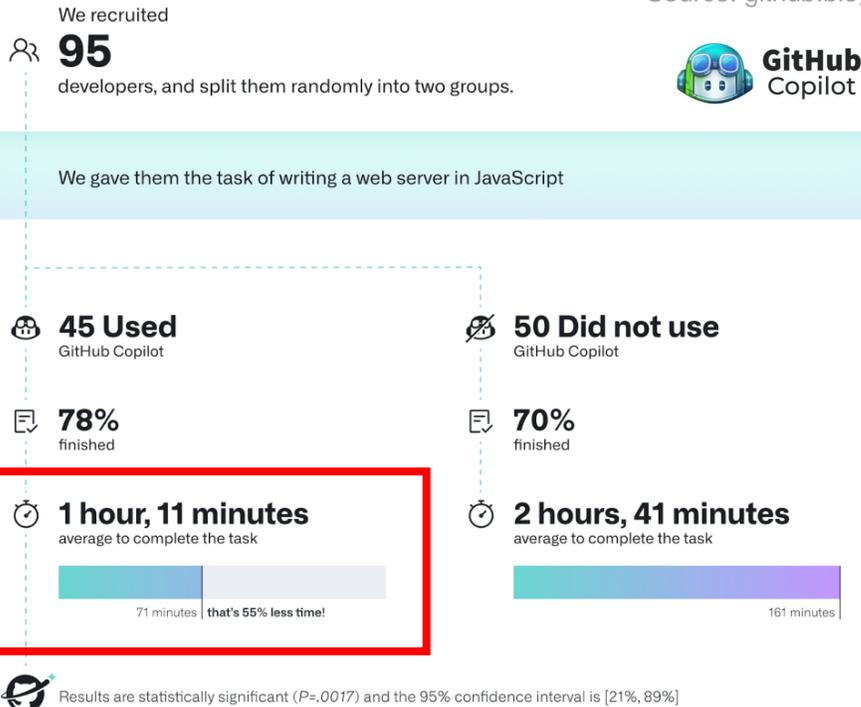
Seven years after the rise of LLM architecture [1], and years after the start of the “chatbot revolution” [2], significant investments have been made to AI-enhance the software developer space. Since the introduction of GitHub Copilot [3], numerous developer tool code editing and generation support have been built for the general developer community [4], [5], [6], [7]. Researchers and educators have also developed prototype tools to assist programmers and students. Furthermore, substantial investments have been made in building tools for internal use, such as Meta [8] and Google [9], [10], [11].

However, there is still much to investigate to determine how useful these tools are in helping developers, speed up their productivity. Truly understanding the productivity benefits of AI enhanced coding tools in the nascent field. While some research has shown improvements in coding speed [12], developer throughput [13], and productivity [14], more work must be done to validate these assertions across, for example, tasks, developer cohorts, and more.

To date, very few estimates of the impact of AI developer tools on time spent on task in an enterprise have been published. One much-discussed study is a randomized controlled trial by Peng et al. [12] ($n =$

Copilot—an AI code assistant—, compared to those not using it. Another enterprise-specific estimate comes from a pooled analysis of three field experiments ($n = 4,867$) conducted by Cui et al. [13], where developers either had access to Copilot or did not have access to it in their daily activities. The authors found a 26% increase in throughput (measured as an increase in the number of pull requests) for developers using Copilot. Although these studies provide valuable insights and help quantify the speed improvements offered by one AI-enhanced developer tool, gaps remain in estimating the overall impact of

Source: github.blog



From **-22%**
to **-55%**
Time to task

Large Language Models for Software Engineering: A Systematic Literature Review

XINYI HOU*, Huazhong University of Science and Technology, China

YANJIE ZHAO*, Huazhong University of Science and Technology, China

YUE LIU, Monash University, Australia

ZHOU YANG, Singapore

KAILONG WANG, Huazhong University of Science and Technology, China

LI LI, Beihang University, China

XIAPU LUO, The Hong Kong Polytechnic University, China

Hou et al. 2023

Large Language Models for Test-Free Fault Localization

Aidan Z.H. Yang
aidan@cmu.edu

Claire Le Goues
clegoues@cs.cmu.edu

Carnegie Mellon University
Pittsburgh, United States

Carnegie Mellon University
Pittsburgh, United States

Ruben Martins
rubem@cs.cmu.edu

Vincent Hellendoorn
vhellendoorn@cmu.edu

Carnegie Mellon University
Pittsburgh, United States

Carnegie Mellon University
Pittsburgh, United States

Yang et al. 2024

ABSTRACT

Fault Localization (FL) aims to automatically localize buggy lines of code, a key first step in many manual and automatic debug tasks. Previous FL techniques assume the provision of input test and often require extensive program analysis, program instrumentation, or data preprocessing. Prior work on deep learning for FL struggles to learn from small datasets and produces limited results on real-world programs. Inspired by the ability of large language models (LLMs) of code to adapt to new tasks based on very few examples, we investigate the applicability of LLMs to line level fault localization. Specifically, we propose to overcome the left-right nature of LLMs by fine-tuning a small set of bidirectional adapter layers on top of the representations learned by LLMs to produce *LLMAO*, the first language model based fault localization approach that locates buggy lines of code without any test coverage information. We fine-tune LLMs with 350 million, 6 billion, and billion parameters on small, manually curated corpora of bug programs such as the *Defects4J* corpus. We observe that our technique achieves substantially more confidence in fault localization when built on the larger models, with bug localization performance scaling consistently with the LLM size. Our empirical evaluation shows that *LLMAO* improves the Top-1 results over the state-of-the-art machine learning fault localization (MLFL) baselines 2.3%-54.4%, and Top-5 results by 14.4%-35.6%. *LLMAO* is also the first FL technique trained using a language model architecture that can detect security vulnerabilities down to the code line level.

CCS CONCEPTS

• Software and its engineering → Software functional properties; • Computing methodologies → Neural networks.

ACM Reference Format:

Aidan Z.H. Yang, Claire Le Goues, Ruben Martins, and Vincent J. Hellendoorn. 2024. Large Language Models for Test-Free Fault Localization. 2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE '24), April 14–20, 2024, Lisbon, Portugal. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3597503.3623342>



This work is licensed under a Creative Commons Attribution International 4.0 License [ICSE '24, April 14–20, 2024, Lisbon, Portugal]. © 2024 Copyright held by the owner(s). ACM ISBN 978-1-4001-6217-4/24/04. <https://doi.org/10.1145/3597503.3623342>

1 INTRODUCTION

Fault localization (FL) [1–4] approaches aim to automatically identify

Using an LLM to Help With Code Understanding

Daye Nam

Andrew Macvean

Vincent Hellendoorn

Carnegie Mellon University
U.S.A.

Google Inc.
U.S.A.

Carnegie Mellon University
U.S.A.

dayenam@cs.cmu.edu

andrewmacvean@google.com

hellend@andrew.cmu.edu

nam@cmu.edu

macvean@cmu.edu

hellend@cmu.edu

nam@cmu.edu

macvean@cmu.edu

hellend@cmu.edu

ABSTRACT

Understanding code is challenging, especially when working in new and complex development environments. Code comments and documentation can help, but are typically scarce or hard to navigate. Large language models (LLMs) are revolutionizing the process of writing code. Can they do the same for helping understand it? In this study, we provide a first investigation of an LLM-based conversational UI built directly in the IDE that is geared towards code understanding. Our IDE plugin queries OpenAI's GPT-3.5-turbo model with four high-level requests without the user having to write explicit prompts to explain a highlighted section of code, provide details of API calls used in the code, explain key domain-specific terms, and provide usage examples for an API. The plugin also allows for open-ended prompts, which are automatically contextualized to the LLM with the program being edited. We evaluate this system in a user study with 32 participants, which confirms that using our plugin can aid task completion more than web search. We additionally provide a thorough analysis of the ways developers use, and perceive the usefulness of, our system, among others finding that the usage and benefits differ between students and professionals. We conclude that in-IDE prompt-less interaction with LLMs is a promising future direction for tool builders.

ACM Reference Format:

Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. 2024. Using an LLM to Help With Code Understanding. In 2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE '24), April 14–20, 2024, Lisbon, Portugal. ACM, New York, NY, USA, 13 pages. <https://doi.org/10.1145/3597503.3639187>

1 INTRODUCTION

Building and maintaining software systems requires a deep understanding of a codebase. Consequently, developers spend a significant amount of time searching and foraging for the information they need and organizing and digesting the information they

find [30, 31, 34, 44, 48, 57]. Understanding code, however, is a challenging task; developers need to assimilate a large amount of information about the semantics of the code, the intricacies of the APIs used, and the relevant domain-specific concepts. Such information is often scattered across multiple sources, making it challenging for developers, especially novices or those working with unfamiliar APIs, to locate what they need. Furthermore, much of the relevant information is inadequately documented or spread across different formats and mediums, where it often becomes outdated.

With the growing popularity of large language model (LLM) based code generation tools [26, 54, 67], the need for information support for code understanding is arguably growing even higher. These tools can generate code automatically, even for developers with limited coding skills or domain knowledge. This convenience comes at a cost, however – developers may receive code they don't understand [24, 79]. Indeed, early research on LLM code generation tools has found that developers have a harder time debugging code generated by the LLM and easily get frustrated [40, 71].

Fortunately, LLMs also provide an opportunity in this space, namely by offering on-demand generation-based information support for developers faced with unfamiliar code. Compared to general web search queries [74], LLM prompts can allow developers to provide more context, which can enable them to receive information that more precisely aligns with their specific needs, potentially reducing the time spent on sifting through the information obtained from the web to suit their particular requirements. Developers have indeed taken to web-hosted conversational LLM tools, such as ChatGPT, for programming support en masse, but this setup requires them to both context switch and copy the relevant context from their IDEs into the chat system for support.

To explore the potential for generation-based information support directly in the developer's programming environment, we developed a prototype in-IDE LLM information support tool, GILT (Generation-based Information-support with LLM Technology). GILT is capable of generating on-demand information while considering the user's local code context, which we incorporate into the



Code generation



Code understanding



Requirements engineering



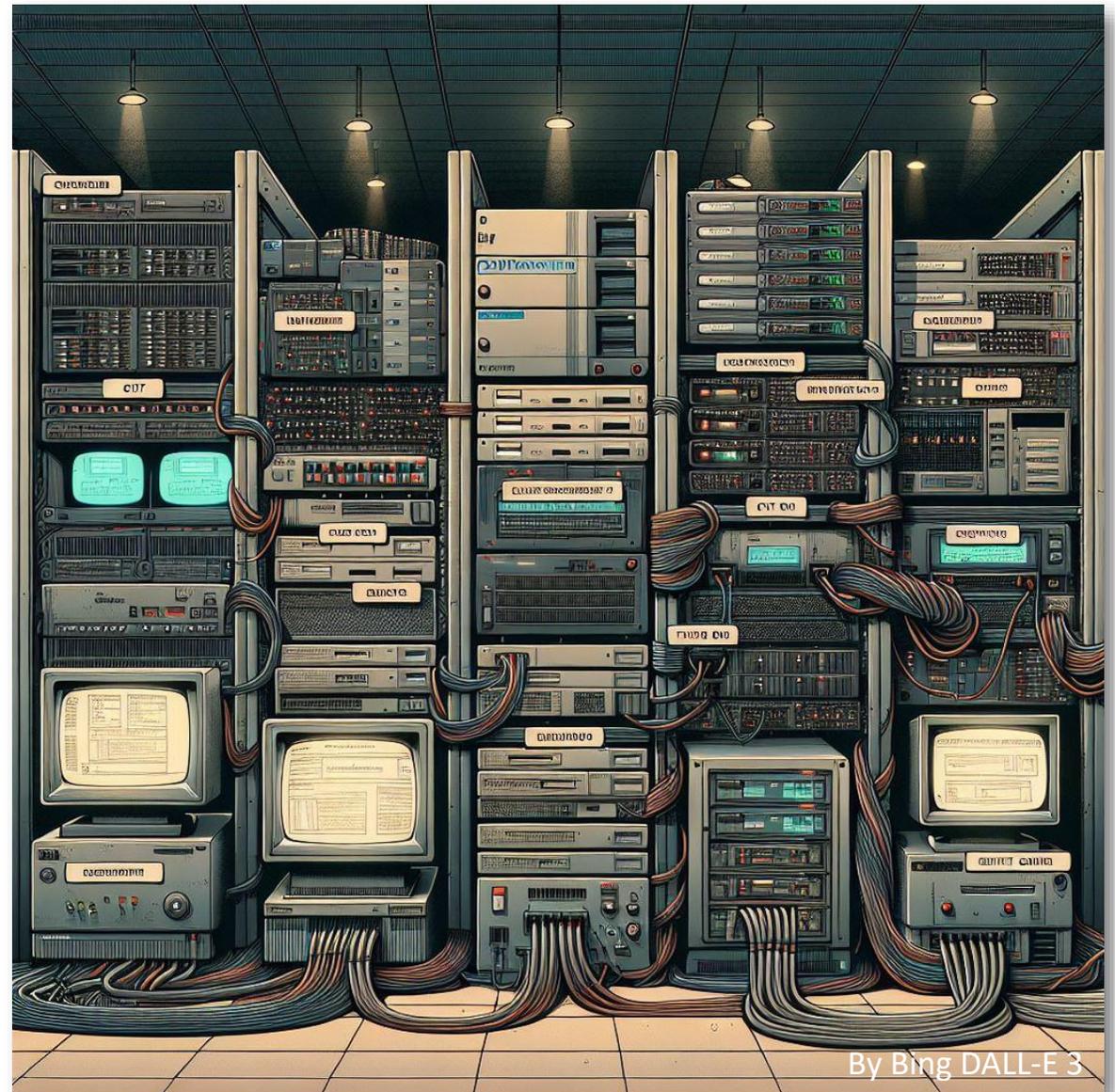
Bug-fixing

... and more

Legacy Systems

Old system or application
that is still in use

Business-critical for the
company



By Bing DALL-E 3

Legacy Systems



Higher
maintenance costs

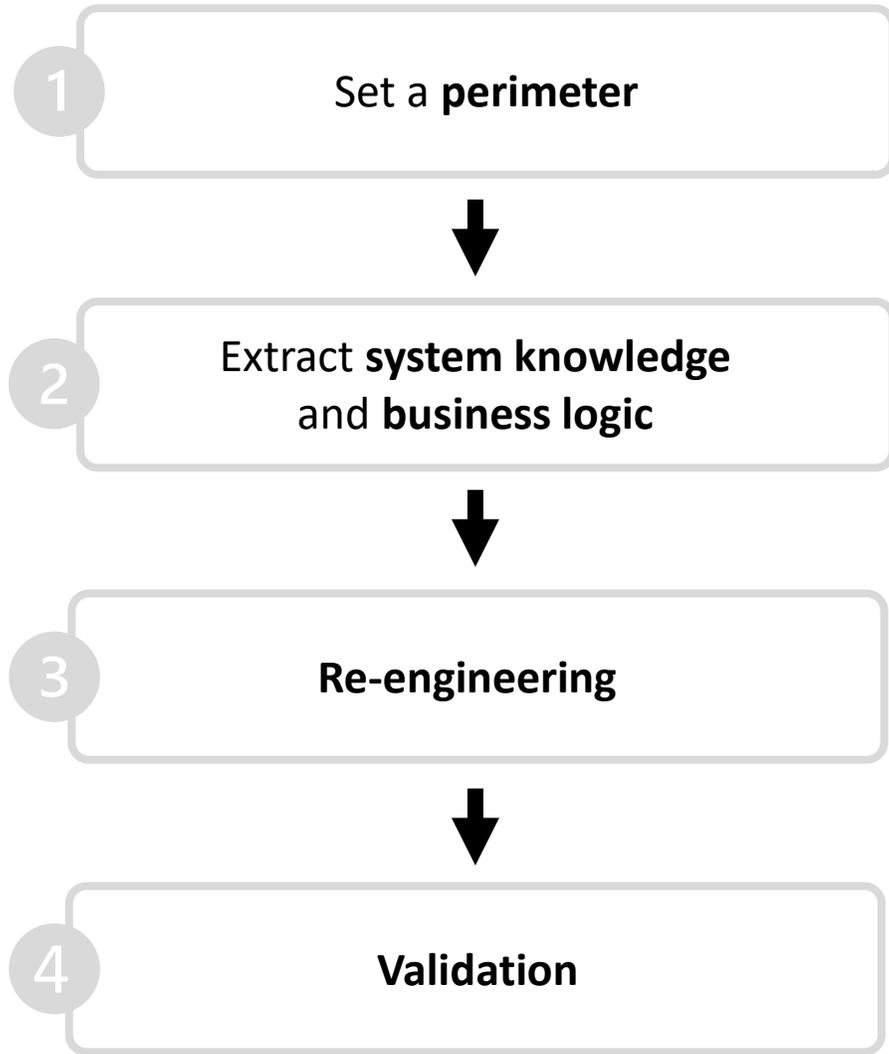


Difficult to extend to new
business requirements

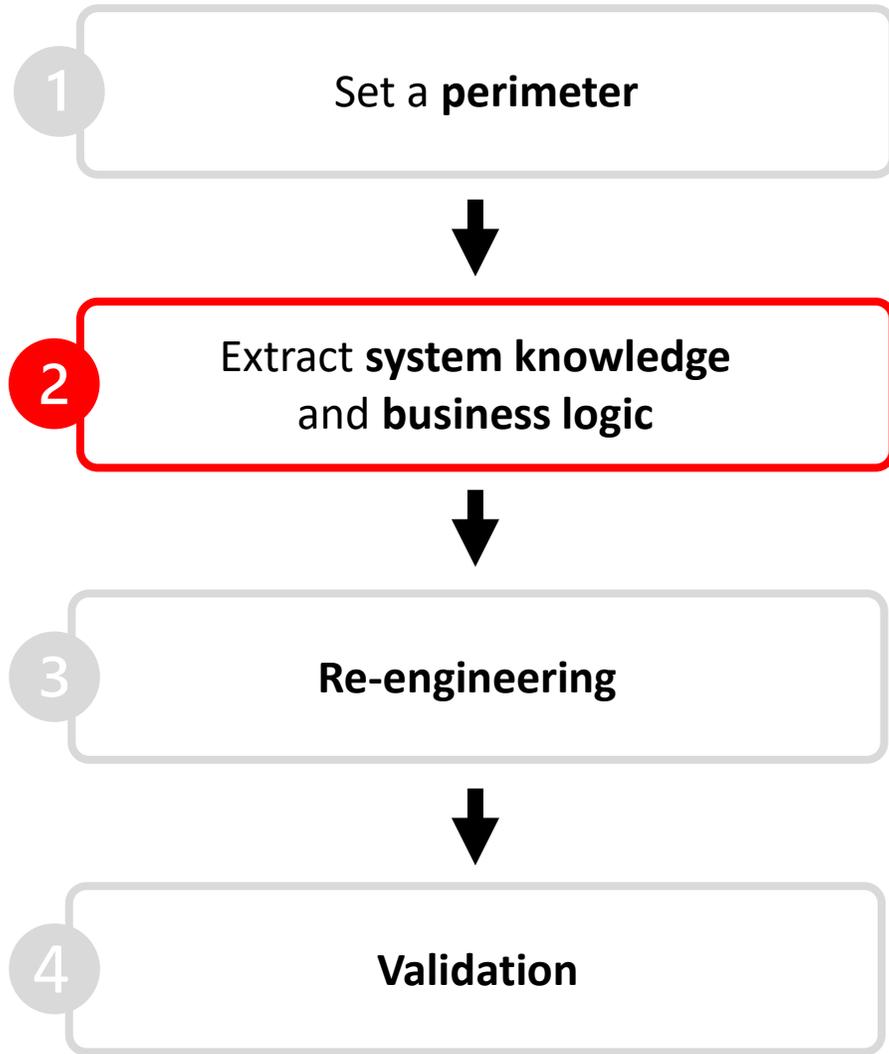


Proneness to **quality**
and **reliability issues**

Driver for legacy system modernization

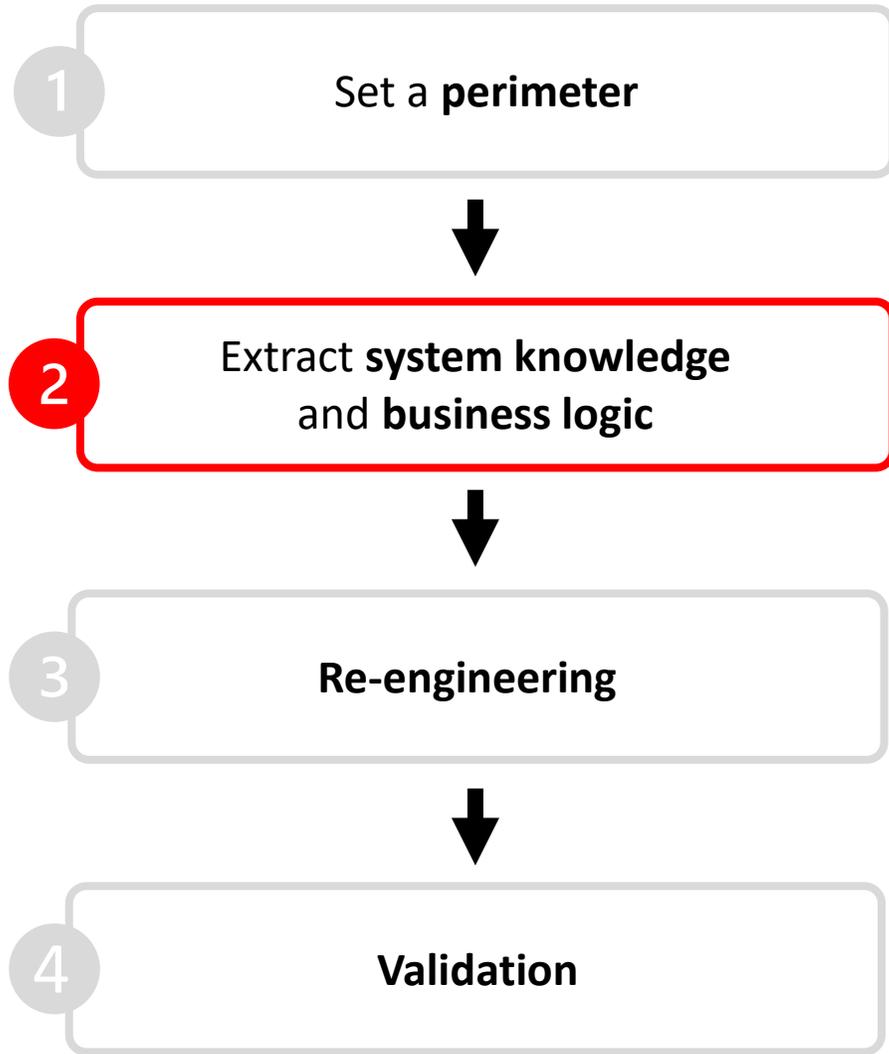


Modernizing Legacy Code



Extracting the encapsulated
system and business knowledge
is hard

Modernizing Legacy Code



Extracting the encapsulated
system and business knowledge
is hard

GenAI can help!

Modernizing Legacy Code

Source Code Summarization in the Era of Large Language Models

Weisong Sun^{1,2}, Yun Miao¹, Yuekang Li³, Hongyu Zhang⁴, Chunrong Fang¹,
Yi Liu*, Gelei Deng*, Yanli Liu*, Zhenyu Chen*

¹State Key Laboratory of Software Development and Environment, Shanghai University, Shanghai, China

²College of Computer Science and Engineering, Singapore University of Technology and Design, Singapore

³School of Computer Science and Engineering, University of New South Wales, Sydney, Australia

⁴School of Big Data and Software Engineering, Chongqing University, Chongqing, China

weisong.sun@ntu.edu.sg, miaoyun001my@gmail.com, yuekang.li@unsw.edu.au, hyzhang@cqu.edu.cn,

fangchunrong@nju.edu.cn, yi009@c.ntu.edu.sg, gelei.deng@ntu.edu.sg, yangliu@ntu.edu.sg, zychen@nju.edu.cn

Sun et al. ICSE 2025

Abstract—To support software developers in understanding and maintaining programs, various automatic (source) code summarization techniques have been proposed to generate a concise natural language summary (i.e., comment) for a given code snippet. Recently, the emergence of large language models (LLMs) has led to a great boost in the performance of code-related tasks. In this paper, we undertake a systematic and comprehensive study on code summarization in the era of LLMs, which covers multiple aspects involved in the workflow of LLM-based code summarization. Specifically, we begin by examining prevalent automated evaluation methods for assessing the quality of summaries generated by LLMs and find that the results of the GPT-4 evaluation method are most closely aligned with human evaluation. Then, we explore the effectiveness of five prompting techniques (zero-shot, few-shot, chain-of-thought, critique, and expert) in adapting LLMs to code summarization tasks. Contrary to expectations, advanced prompting techniques may not outperform simple zero-shot prompting. Next, we investigate the impact of LLMs’ model settings (including `top_p` and temperature parameters) on the quality of generated summaries. We find the impact of the two parameters on summary quality varies by the base LLM and programming language, but their impacts are similar. Moreover, we canvass LLMs’ abilities to summarize code snippets in distinct types of programming languages. The results reveal that LLMs perform suboptimally when summarizing code written in logic programming languages compared to other language types (e.g., procedural and object-oriented programming languages). Finally, we unexpectedly find that CodeLlama-Instruct with 7B parameters can outperform advanced GPT-4 in generating summaries describing code implementation details and asserting code properties. We hope that our findings can provide a comprehensive understanding of code summarization in the era of LLMs.

Index Terms—large language model, source code summarization, prompt engineering

I. INTRODUCTION

Code comments are vital for enhancing program comprehension [1] and facilitating software maintenance [2]. While it is considered good programming practice to write high-quality comments, the process is often labor-intensive and time-consuming [2]–[4]. As a result, high-quality comments are frequently absent, mismatched, or outdated during software evolution, posing a common problem in the software industry [5]–[8]. Automatic code summarization (or simply,

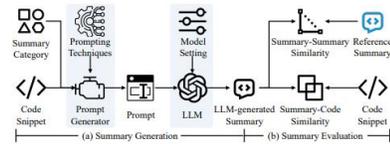


Fig. 1: General workflow of LLM-based code summarization and its effectiveness evaluation

code summarization), a hot research topic [9]–[12], addresses this challenge by developing advanced techniques/models for automatically generating natural language summaries (i.e., comments) for code snippets, such as Java methods or Python functions, provided by developers.

Recently, with the success of large language models (LLMs) in natural language processing (NLP) [13], [14], an increasing number of software engineering (SE) researchers have started integrating them into the resolution process of various SE tasks [15]–[18], such as code generation [19], [20], program repair [21], [22], and vulnerability detection/localization [23], [24]. In this study, we focus on the application of LLMs on the code summarization tasks. Figure 1 shows the general workflow of LLM-based code summarization and its effectiveness evaluation methods. In the summary generation process, the input consists of a code snippet and the expected summary category. The input is passed to a prompt generator equipped with various prompt engineering techniques (referred to as prompting technique), which constructs a prompt based on input. This prompt is then used to instruct LLMs to generate a summary of the expected type for the input code snippet. In the summary evaluation process, a common method used to automatically assess the quality of LLM-generated summaries is to compute the text or semantic similarity between the LLM-generated summaries and the reference summaries.

There have been several recent studies investigating the effectiveness of LLMs in code summarization tasks [25]–[29]. These studies can help subsequent researchers rapidly under-

LLM-based Code Summarization

Extract business logic and code knowledge

SYS PROMPT:

You are a code documentation assistant.
Explain in a few words the purpose of the input code snippet for a non-technical person.

+

```
1 Class Utils.Check [ Abstract ] {
2
3 ClassMethod Luhn(x As %String) As %Boolean {
4     SET x=$TRANSLATE(x," "), cd=$EXTRACT(x,*)
5     SET x=$REVERSE($EXTRACT(x,1,*-1)), t=0
6     FOR i=1:1:$LENGTH(x) {
7         SET n=$EXTRACT(x,i)
8         IF i#2 SET n=n*2 IF $LENGTH(n)>1 SET n=$EXTRACT(n,1)+$EXTRACT(n,2)
9         SET t=t+n
10    }
11    QUIT cd=((t*9)#10)
12 }
13
14 }
```

=

This code is designed to check the validity of a number using the **Luhn algorithm**, which is a simple method often used to verify credit card numbers.

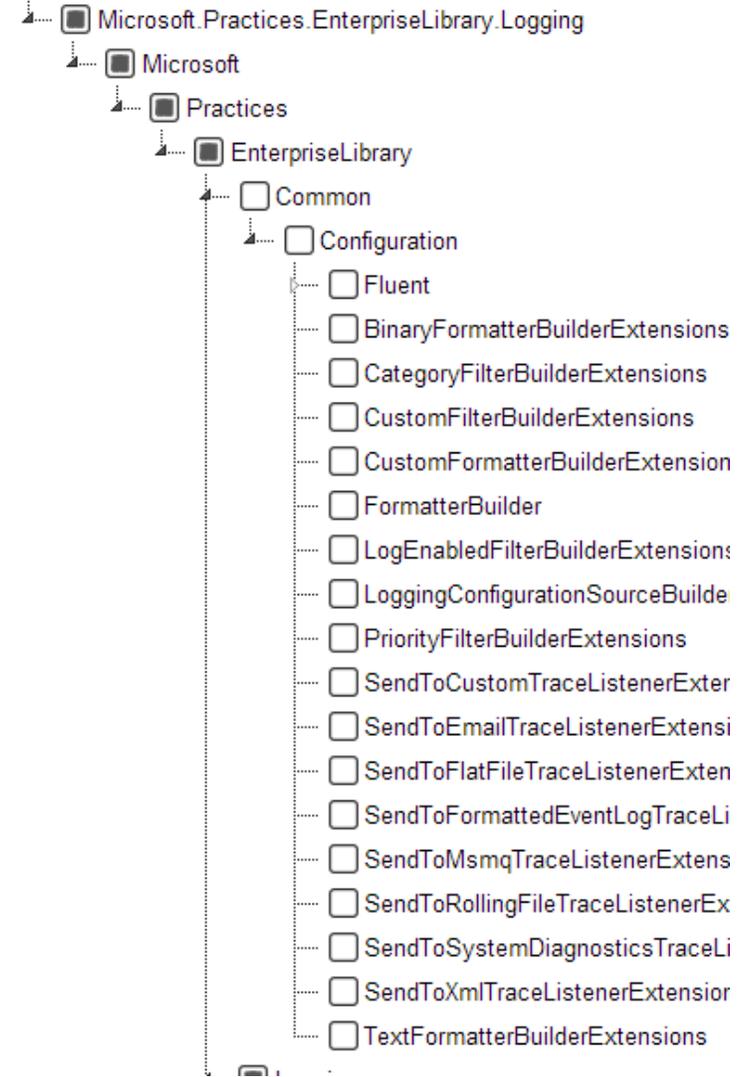
It processes the number provided and computes a check digit, ensuring that the number is correctly formatted for purposes like fraud prevention.



Code Summarization: Example



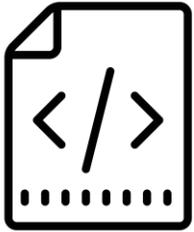
ANALYZE ANOTHER ASSEMBLY



Source: gmamaladze.wordpress.com

How to summarize complex codebases ?

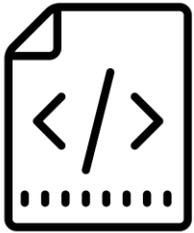
1



Select input
sources

Summarizing a complex codebase

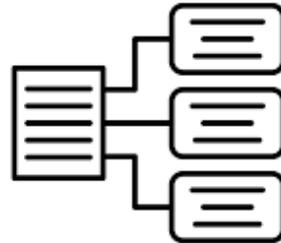
1



Select input
sources



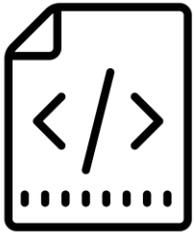
2



Extract sources
structure and
dependencies

Summarizing a complex codebase

1



Select input sources



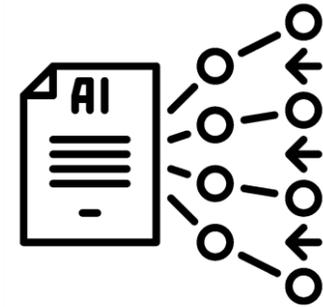
2



Extract sources structure and dependencies



3



Generate natural language summary

Summarizing a complex codebase

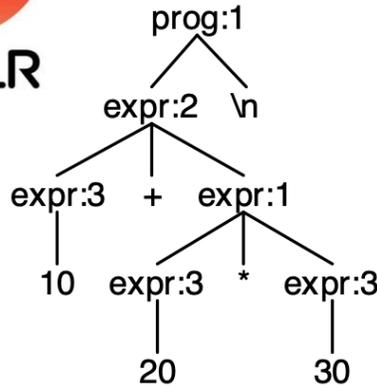


Codebase
subset

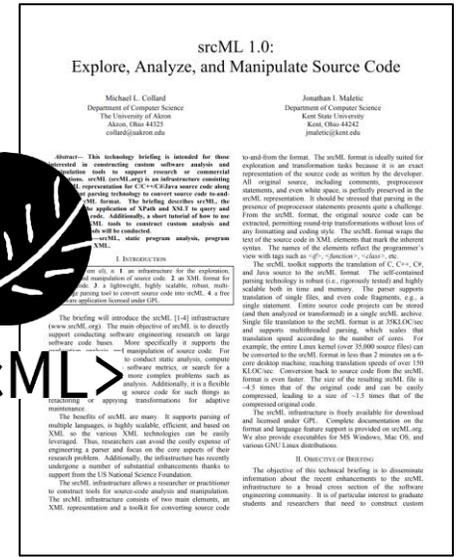
InterSystems
CACHE



Language-native
exporting tools



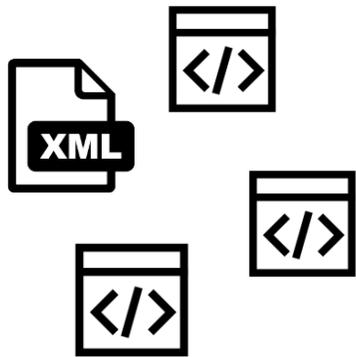
Custom
code parsers



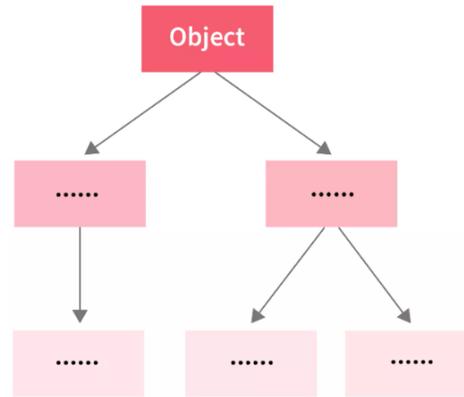
Existing
code analysis tools

1

Select input sources



Codebase subset



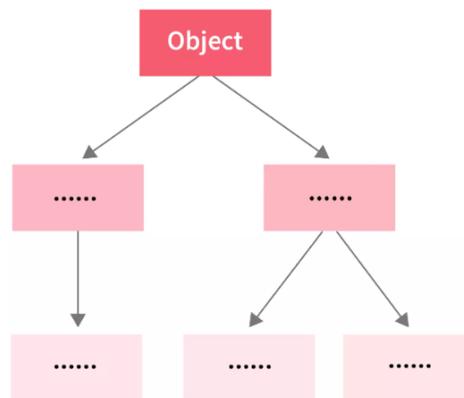
Code dependency graph



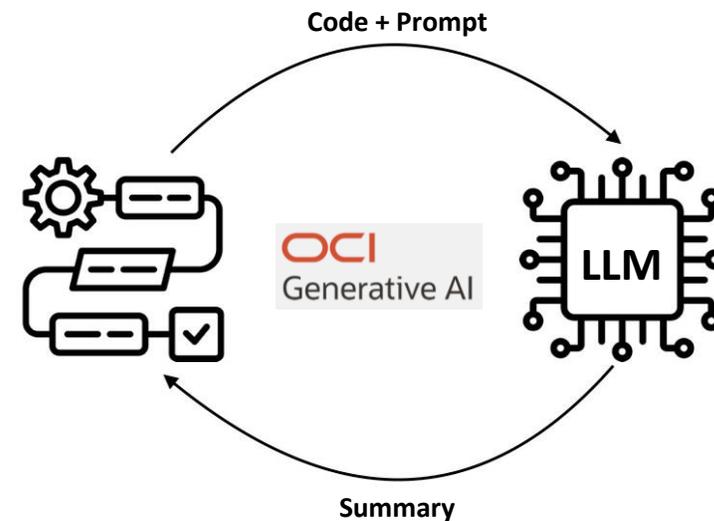
2 Build codebase structure and dependencies



Codebase subset



Code dependency graph



Iterative prompting

```

1 Class Hospital.PatientAdmission Extends %Persistent
2
3 // Properties
4 Property AdmissionID As %String [ InitialExpression = {$System.Encryption.GenCryptRand(8)} ];
5 Property PatientID As %String;
6 Property AdmissionDate As %Date;
7 Property DischargeDate As %Date;
8 Property RoomNumber As %String;
9 Property AssignedDoctorID As %String;
10 Property Procedures As list Of %String;
11 Property InsuranceApproved As %Boolean [ InitialExpression = 0 ];
12 Property AdmissionStatus As %String(VALUELIST="Pending,Active,Discharged,Cancelled") [ InitialExpression = "Pending" ];
13 Property CancellationReason As %String;
14
15 // Index based on AdmissionID
16 Index AdmissionIndex On AdmissionID [ IdKey ];
17
18 // Use %OpenId with AdmissionID
19 Parameter IDPROPERTY = "AdmissionID";
20
21 // Method to initiate the admission process
22 Method InitiateAdmission(patientID As %String, doctorID As %String) As %Status
23 {
24 // Step 1: Validate inputs
25 If patientID = "" || doctorID = "" {
26     Return $$$ERROR($$$GeneralError, "Patient ID and Doctor ID are required.")
27 }
28
29 // Step 2: Check if patient exists
30 Set patient = ##class(Hospital.Patient).%OpenId(patientID)
31 If 'patient {
32     Return $$$ERROR($$$GeneralError, "Patient not found.")
33 }
34
35 // Step 3: Assign initial values
36 Set ..PatientID = patientID
37 Set ..AssignedDoctorID = doctorID
38 Set ..AdmissionDate = %ZDATE(%SHOROLOG,8)
39 Set ..Procedures = ##class(%ListOfDataTypes).%New()
40 Set ..AdmissionStatus = "Pending"
41
42 // Step 4: Request insurance approval
43 Set status = ..RequestInsuranceApproval()
44 If $$$ISERR(status) {
45     Return status
46 }
47
48 // Step 5: Assign a room
49 Set status = ..AssignRoom()
50 If $$$ISERR(status) {
51     Return status
52 }
53
54 // Step 6: Update admission status
55 Set ..AdmissionStatus = "Active"
56
57 // Step 7: Save the admission record
58 Set status = ..%Save()
59 If $$$ISERR(status) {
60     Return status
61 }
62
63 Write "Admission initiated successfully. Admission ID: ", ..AdmissionID, !
64 Quit $$$OK
65 }

```

Medical clinic management system

e.g. patient admission, dismissions, medical treatments, allergies, etc.

(Written in **Caché ObjectScript**) 

Example Scenario

Setup authentication

Prompts

LLM parameters

```
1 from llama_index.llms.oci_genai import OCIGenAI
2 from llama_index.core.llms import ChatMessage
3
4
5 # load the OCI config using the default profile
6 config = oci.config.from_file('~/.oci/config', "DEFAULT")
7 # ensure the config is valid
8 oci.config.validate_config(config)
9
10 # Define an example conversation
11 messages = [
12     ChatMessage(role="system", content="You are an AI assistant providing information on Oracle services. Limit your
13     responses to a few sentences."),
14     ChatMessage(role="user", content="Tell me something about Oracle Cloud Infrastructure."),
15 ]
16 # Initialize the LLM
17 llm = OCIGenAI(
18     model="cohere.command-r-16k", # e.g., "cohere.command-r-plus", "meta.llama-3.1-70b-instruct"
19     service_endpoint="https://inference.generativeai.eu-frankfurt-1.oci.oraclecloud.com", # EU service endpoint
20     compartment_id="<YOUR.OCI.COMPARTMENT.ID", # your compartment ID from OCI
21     temperature=0.7
22 )
23
24 # Run inference
25 response = llm.chat(messages)
26 print(response)
```

✓ [16] 1s 607ms

```
assistant: Oracle Cloud Infrastructure is a suite of cloud computing services that provides the foundation for >
<customers to build, deploy, and run applications in the Oracle Cloud. It offers a wide range of global compute >
<resources, storage options, and high-speed networking designed to ensure reliability, scalability, and security. >
<Oracle Cloud Infrastructure also integrates seamlessly with other Oracle Cloud applications.
```



1

Setup the LLM inference

```
1 system_prompt_class = """
2 Impersonate a Caché ObjectScript software engineer.
3
4 # **Steps**
5 You task is to help me, a developer with no expertise in Caché ObjectScript, to understand the functionalities of the input class.
6 Provide a brief summary in two or three sentences describing its purpose.
7 Write the summary similarly to a Javadoc class description, summarizing the embedded business logic.
8
9 # **Output**
10
11 # Class: <class name>
12
13 <summary>
14
15 (Start the summary like this: "The class <class name> ...")
16 """
```

2

Extract **implementation logic**
from methods

```
1 system_prompt_method = """
2 Impersonate a Caché ObjectScript software engineer.
3
4 # **Steps**
5 Step by step, describe in brief what the provided method implementation does in such a way a person without domain knowledge
6 of the entire codebase can understand and reimplement it in another programming language.
7
8 # **Output**
9 Format the report as follows:
10
11 ## <class>: <Method name>
12
13 ### Method signature
14 - Parameters: <method parameters along the type>
15 - Return type: <method return type>
16
17 ### Method functionality
18 - <bullet>
19 - <bullet>
20 ...
21
22 (The number of bullets should be proportional to the complexity of the method)
23 """
```

1

Extract **high-level description**
of class functionalities

2

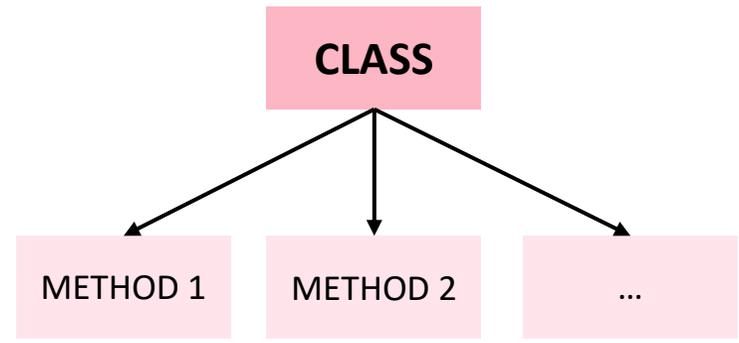
Define prompts for summarization

```

1 Class Hospital.PatientAdmission Extends %Persistent
2
3 // Properties
4 Property AdmissionID As %String [ InitialExpression = {$System.Encryption.GenCryptRand(8)} ];
5 Property PatientID As %String;
6 Property AdmissionDate As %Date;
7 Property DischargedDate As %Date;
8 Property RoomNumber As %String;
9 Property AssignedDoctorID As %String;
10 Property Procedures As list Of %String;
11 Property InsuranceApproved As %Boolean [ InitialExpression = 0 ];
12 Property AdmissionStatus As %String(VALUELIST="Pending,Active,Discharged,Cancelled") [ InitialExpression = "Pending" ];
13 Property CancellationReason As %String;
14
15 // Index based on AdmissionID
16 Index AdmissionIndex On AdmissionID [ IdKey ];
17
18 // Use %OpenId with AdmissionID
19 Parameter IDPROPERTY = "AdmissionID";
20
21 // Method to initiate the admission process
22 Method InitiateAdmission(patientID As %String, doctorID As %String) As %Status
23 {
24 // Step 1: Validate inputs
25 If patientID = "" || doctorID = "" {
26 Return $$$ERROR($$$GeneralError, "Patient ID and Doctor ID are required.")
27 }
28
29 // Step 2: Check if patient exists
30 Set patient = ##class(Hospital.Patient).%OpenId(patientID)
31 If 'patient {
32 Return $$$ERROR($$$GeneralError, "Patient not found.")
33 }
34
35 // Step 3: Assign initial values
36 Set ..PatientID = patientID
37 Set ..AssignedDoctorID = doctorID
38 Set ..AdmissionDate = %ZDATE(%SHOROLOG,8)
39 Set ..Procedures = ##class(%ListOfDataTypes).%New()
40 Set ..AdmissionStatus = "Pending"
41
42 // Step 4: Request insurance approval
43 Set status = ..RequestInsuranceApproval()
44 If $$$ISERR(status) {
45 Return status
46 }
47
48 // Step 5: Assign a room
49 Set status = ..AssignRoom()
50 If $$$ISERR(status) {
51 Return status
52 }
53
54 // Step 6: Update admission status
55 Set ..AdmissionStatus = "Active"
56
57 // Step 7: Save the admission record
58 Set status = ..%Save()
59 If $$$ISERR(status) {
60 Return status
61 }
62
63 Write "Admission initiated successfully. Admission ID: ", ..AdmissionID, !
64 Quit $$$OK
65 }

```

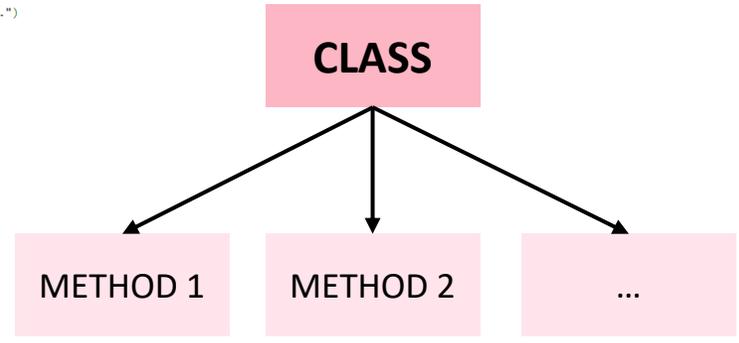
Extract class features



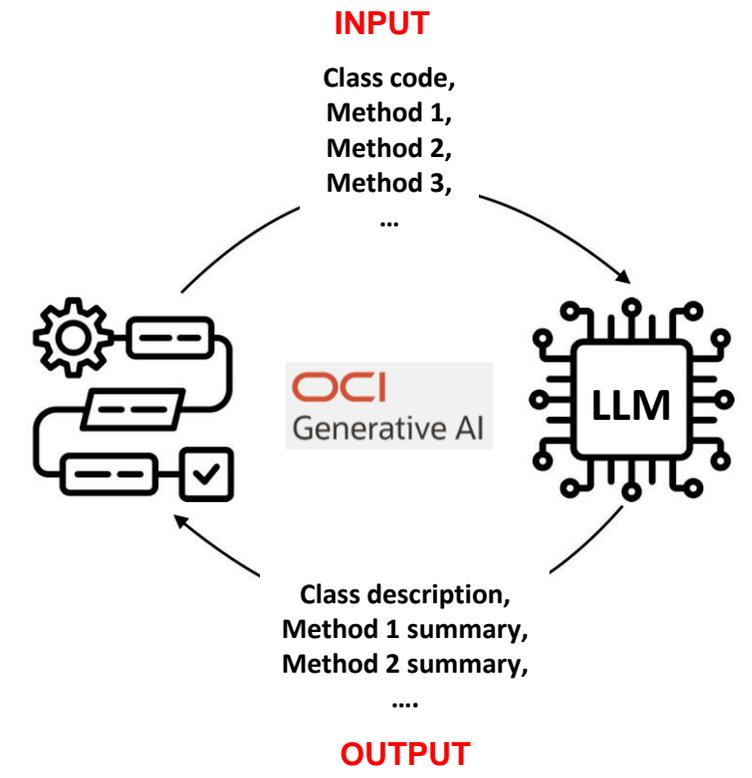
```

1 Class Hospital.PatientAdmission Extends %Persistent
2
3 // Properties
4 Property AdmissionID As %String [ InitialExpression = {$System.Encryption.GenCryptRand(8)} ];
5 Property PatientID As %String;
6 Property AdmissionDate As %Date;
7 Property DischargeDate As %Date;
8 Property RoomNumber As %String;
9 Property AssignedDoctorID As %String;
10 Property Procedures As list Of %String;
11 Property InsuranceApproved As %Boolean [ InitialExpression = 0 ];
12 Property AdmissionStatus As %String(VALUELIST="Pending,Active,Discharged,Cancelled") [ InitialExpression = "Pending" ];
13 Property CancellationReason As %String;
14
15 // Index based on AdmissionID
16 Index AdmissionIndex On AdmissionID [ IdKey ];
17
18 // Use %OpenId with AdmissionID
19 Parameter IDPROPERTY = "AdmissionID";
20
21 // Method to initiate the admission process
22 Method InitiateAdmission(patientID As %String, doctorID As %String) As %Status
23 {
24 // Step 1: Validate inputs
25 If patientID = "" || doctorID = "" {
26     Return $$$ERROR($$$GeneralError, "Patient ID and Doctor ID are required.")
27 }
28
29 // Step 2: Check if patient exists
30 Set patient = ##class(Hospital.Patient).%OpenId(patientID)
31 If 'patient {
32     Return $$$ERROR($$$GeneralError, "Patient not found.")
33 }
34
35 // Step 3: Assign initial values
36 Set ..PatientID = patientID
37 Set ..AssignedDoctorID = doctorID
38 Set ..AdmissionDate = %ZDATE(%HOROLOG,8)
39 Set ..Procedures = ##class(%ListOfDataTypes).%New()
40 Set ..AdmissionStatus = "Pending"
41
42 // Step 4: Request insurance approval
43 Set status = ..RequestInsuranceApproval()
44 If $$$ISERR(status) {
45     Return status
46 }
47
48 // Step 5: Assign a room
49 Set status = ..AssignRoom()
50 If $$$ISERR(status) {
51     Return status
52 }
53
54 // Step 6: Update admission status
55 Set ..AdmissionStatus = "Active"
56
57 // Step 7: Save the admission record
58 Set status = ..%Save()
59 If $$$ISERR(status) {
60     Return status
61 }
62
63 Write "Admission initiated successfully. Admission ID: ", ..AdmissionID, !
64 Quit $$$OK
65 }

```



Iterative prompting



```

1 Class Hospital.PatientAdmission Extends %Persistent
2 {
3     /// Properties
4     Property AdmissionID As %String [ InitialExpression = {$System.Encryption.GenCryptRand(8)} ];
5     Property PatientID As %String;
6     Property AdmissionDate As %Date;
7     Property DischargeDate As %Date;
8     Property RoomNumber As %String;
9     Property AssignedDoctorID As %String;
10    Property Procedures As list Of %String;
11    Property InsuranceApproved As %Boolean [ InitialExpression = 0 ];
12    Property AdmissionStatus As %String(VALUELIST=","Pending,Active,Discharged,Cancelled") [ InitialExpression = "Pending" ];
13    Property CancellationReason As %String;
14
15    /// Index based on AdmissionID
16    Index AdmissionIndex On AdmissionID [ IdKey ];
17
18    /// Use %OpenId with AdmissionID
19    Parameter IDPROPERTY = "AdmissionID";
20
21    /// Method to initiate the admission process
22    Method InitiateAdmission(patientID As %String, doctorID As %String) As %Status
23    {
24        // Step 1: Validate inputs
25        If patientID = "" || doctorID = "" {
26            | Return $$$ERROR($$GeneralError, "Patient ID and Doctor ID are required.")
27        }
28
29        // Step 2: Check if patient exists
30        Set patient = ##class(Hospital.Patient).%OpenId(patientID)
31        If 'patient {
32            | Return $$$ERROR($$GeneralError, "Patient not found.")
33        }
34
35        // Step 3: Assign initial values
36        Set ..PatientID = patientID
37        Set ..AssignedDoctorID = doctorID
38        Set ..AdmissionDate = $ZDATE($HOROLOG,8)
39        Set ..Procedures = ##class(%ListOfDataTypes).%New()
40        Set ..AdmissionStatus = "Pending"
41
42        // Step 4: Request insurance approval
43        Set status = ..RequestInsuranceApproval()
44        If $$$ISERR(status) {
45            | Return status
46        }
47
48        // Step 5: Assign a room
49        Set status = ..AssignRoom()
50        If $$$ISERR(status) {
51            | Return status
52        }
53
54        // Step 6: Update admission status
55        Set ..AdmissionStatus = "Active"
56
57        // Step 7: Save the admission record
58        Set status = ..%Save()
59        If $$$ISERR(status) {
60            | Return status
61        }
62
63        Write "Admission initiated successfully. Admission ID: ", ..AdmissionID, !
64        Quit $$$OK
65    }

```

Class summary

Method description (1)

Method description (2)



Class: Hospital.PatientAdmission

The class Hospital.PatientAdmission represents a patient's admission to a hospital, encapsulating the business logic for managing the admission process, including initiation, scheduling of procedures, discharge, and cancellation. It provides methods for requesting insurance approval, assigning a room, scheduling procedures, discharging the patient, and finalizing billing.

Hospital.PatientAdmission: InitiateAdmission

Method signature

- Parameters:
 - patientID As %String
 - doctorID As %String
- Return type: %Status

Method functionality

- Validates the input parameters `patientID` and `doctorID` to ensure they are not empty.
- Checks if a patient with the provided `patientID` exists in the system.
- Initializes the admission record with the provided `patientID` and `doctorID`, and sets the admission date to the current date.
- Requests insurance approval for the admission.
- Assigns a room to the patient.
- Updates the admission status to "Active".
- Saves the admission record to the database.
- Returns a success status if all steps are completed successfully, or an error status if any step fails.

Hospital.PatientAdmission: FinalizeBilling

Method signature

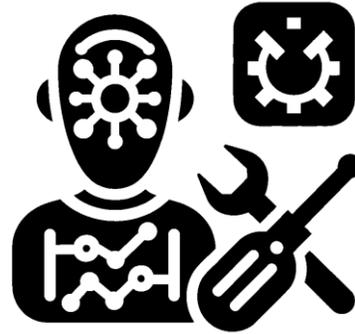
- Parameters: None
- Return type: %Status

Method functionality

- Calculates the total billing amount for a patient's hospital stay by summing up the costs of procedures and room charges.
- Retrieves the costs of procedures by iterating through the patient's procedures and looking up the cost of each procedure in the Hospital.Procedure class.
- Calculates the room charge based on the number of days the patient was admitted, assuming a fixed cost per day.
- Creates a new billing record with the calculated total amount, patient ID, and other relevant details.
- Saves the billing record to the database and returns a status indicating success or failure.



**Limited context
window**



**Model and
prompt tuning**

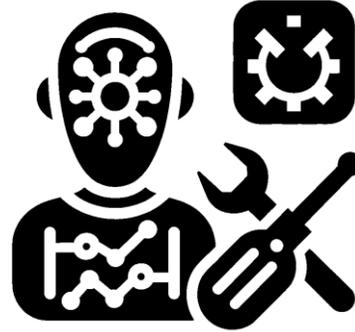


**Code summaries
validation**



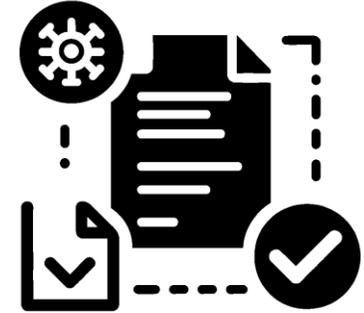
Limited context window

Process artifacts as chunks



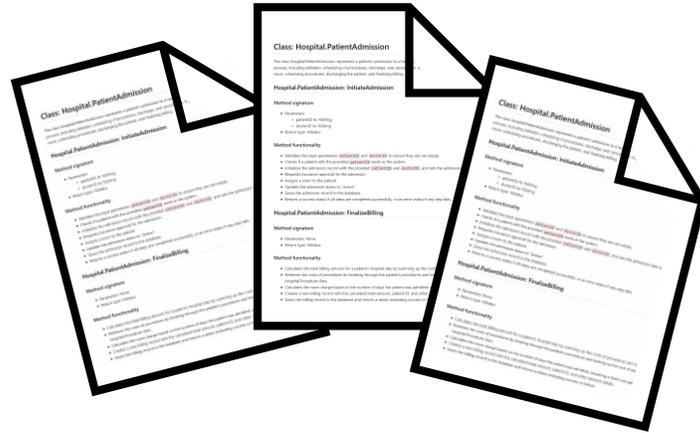
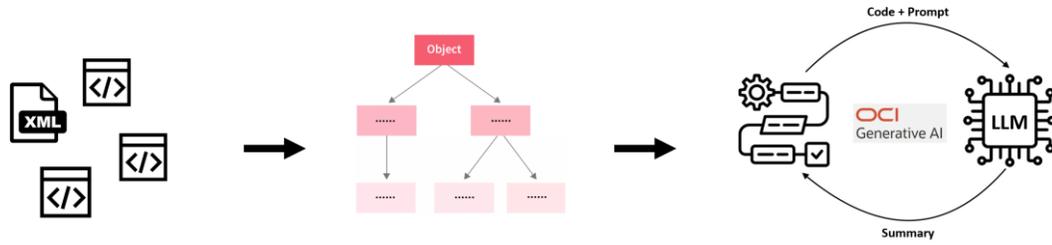
Model and prompt tuning

Context information and LLM costs

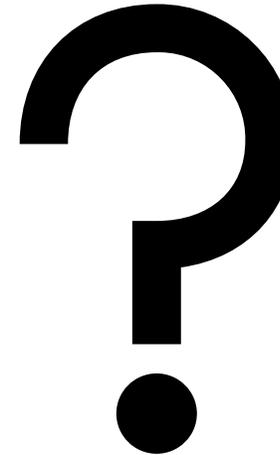


Code summaries validation

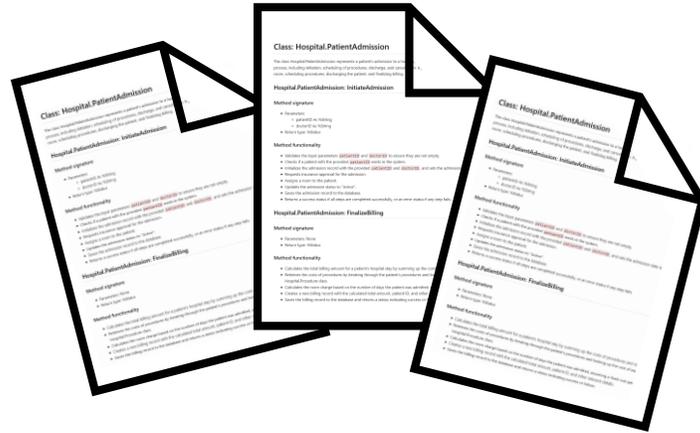
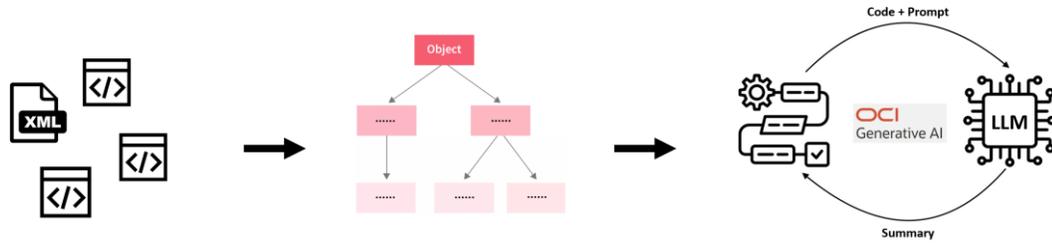
Manual validate a small sample



Extracted codebase knowledge



What's next?



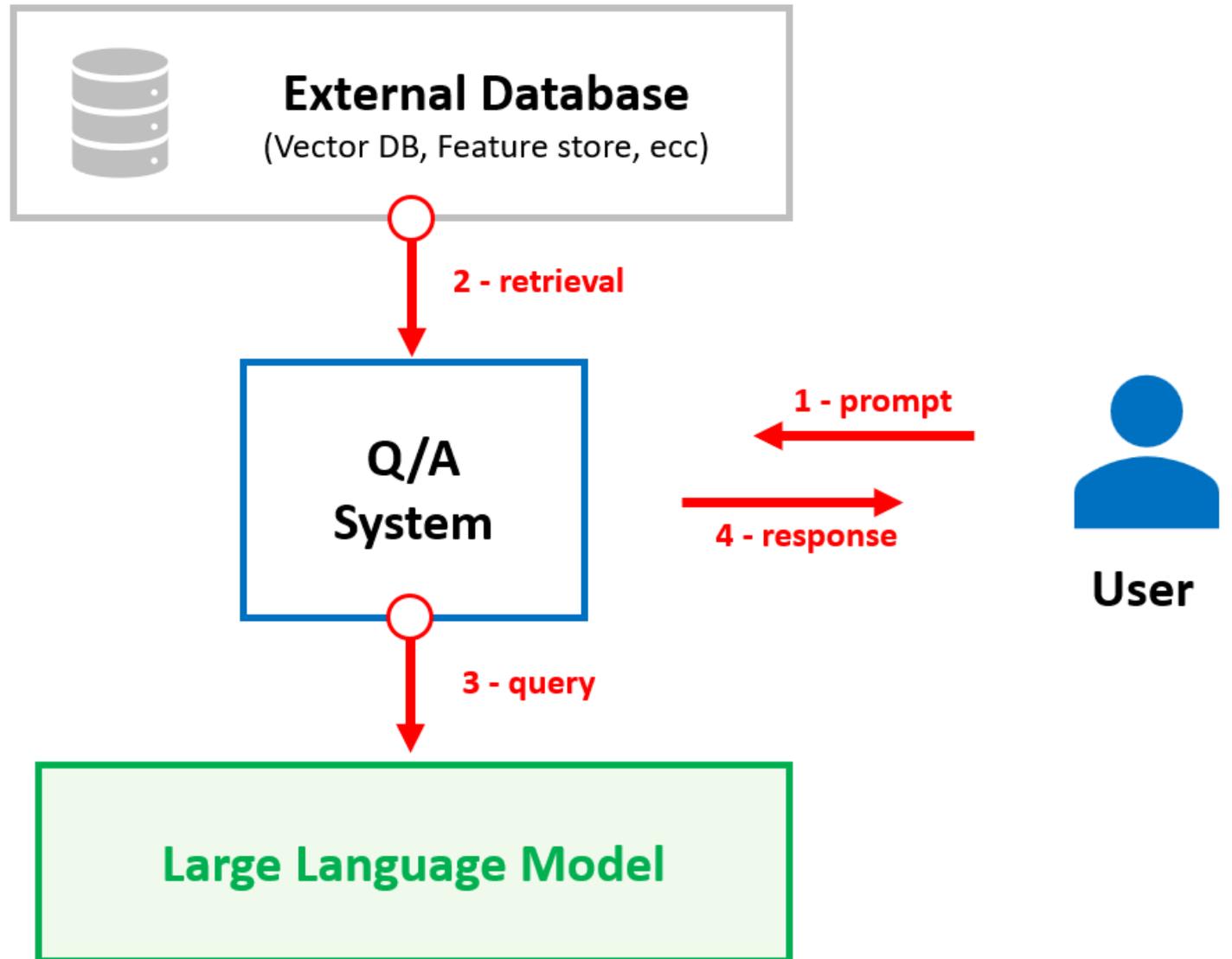
Extracted codebase knowledge

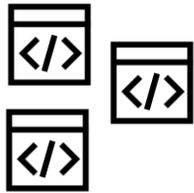
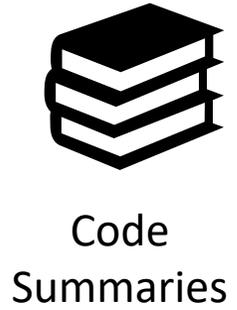


Q/A assistant
with codebase knowledge

What's next?

Retrieval Augmented Generation



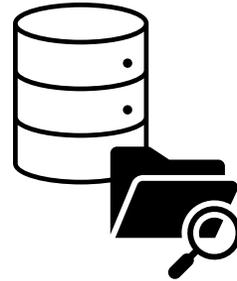


Source code

Vector
Embeddings

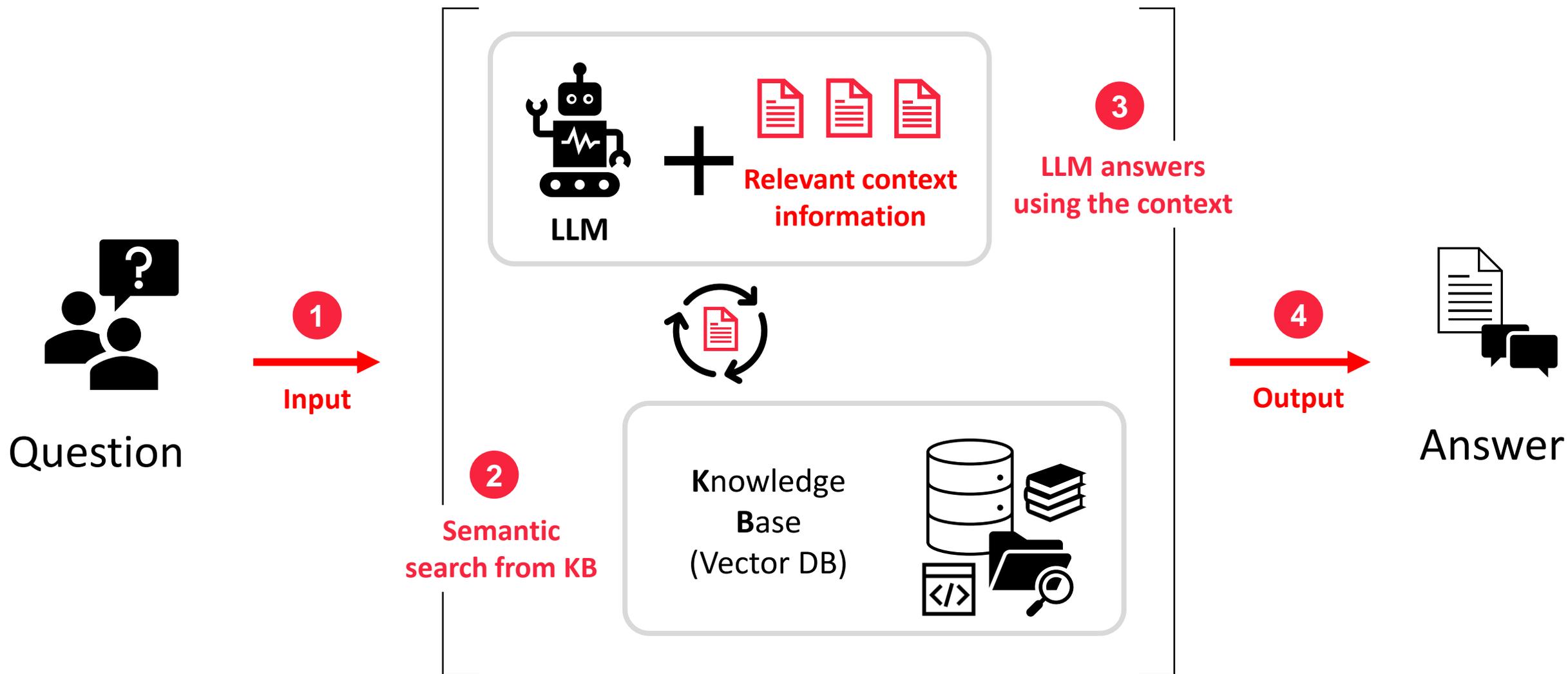


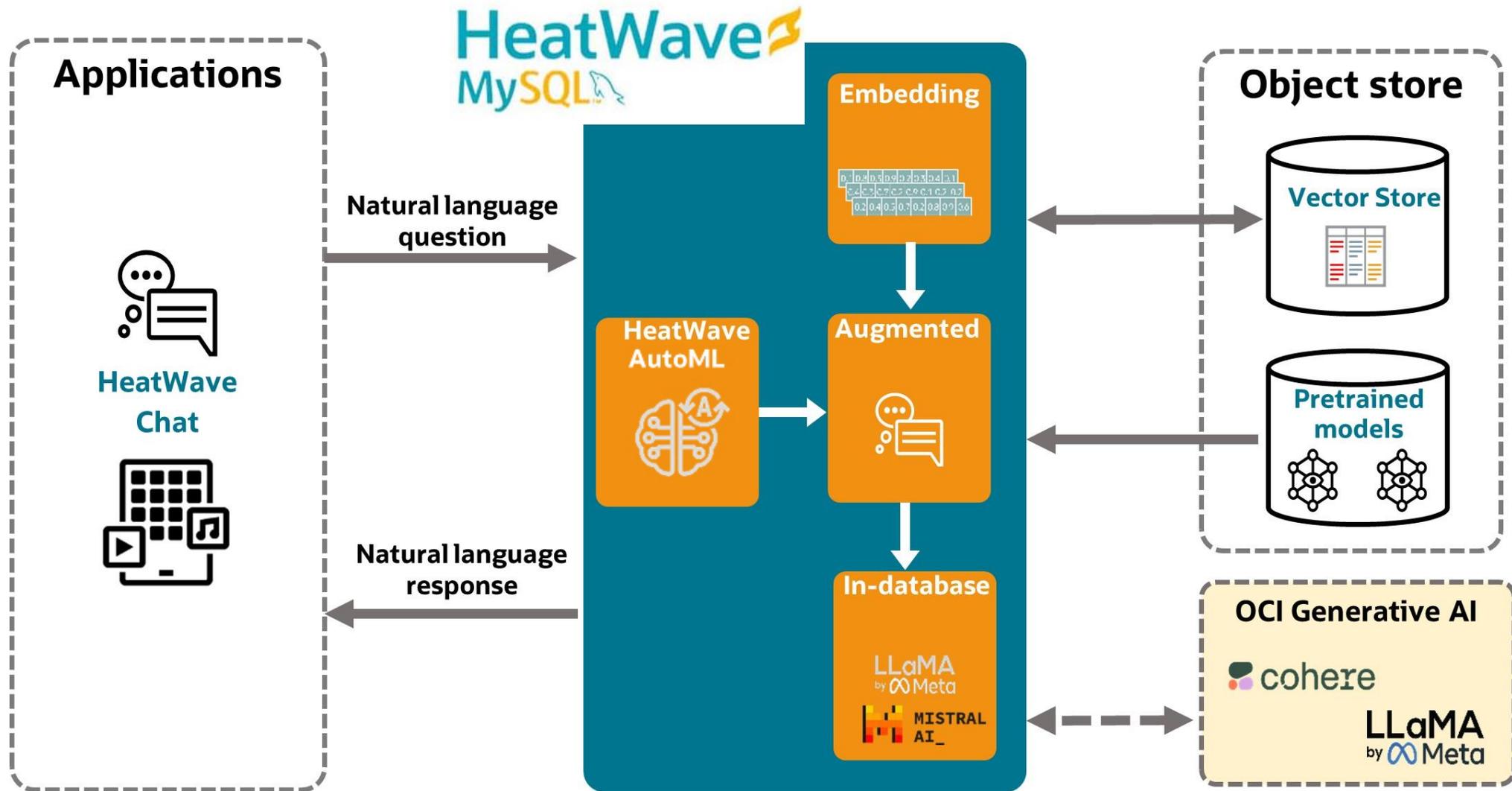
Knowledge
Base
(Vector DB)



1

Data ingestion





Source: blogs.oracle.com

RAG Q/A using MySQL HeatWave

1

Load documents in
OCI Object Storage



2

Select the model
(in-database LLM or OCI GenAI)

```
MySQL>  
CALL sys.ML_MODEL_LOAD("mistral-7b-instruct-v1", NULL);
```

3

Initialize **vector store**

```
MySQL> CALL sys.vector_store_load(  
  "oci://user_bucket@user_namespace/pdf_files/",  
  '{"formats": ["pdf"],  
   "table_name": "pdf_store",  
   "schema_name": "user_documents"}');
```

4

Q/A

```
MySQL>  
CALL sys.ML_RAG("What is this document about?",  
  @output, NULL);
```

RAG Q/A using MySQL HeatWave - example

Summary

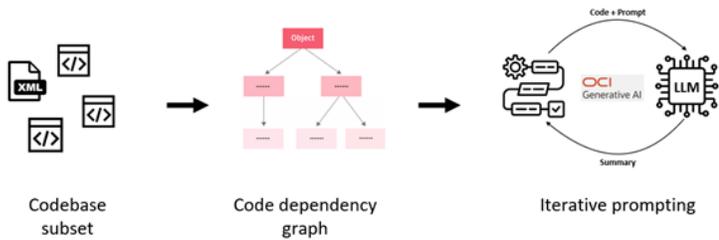
November
2022



Prompt
Engineering



More at: www.promptingguide.ai



3

Extract code summaries



Limited context window

Process artifacts as chunks



Model and prompt tuning

Context information and LLM costs



Code summaries validation

Manual small sample



Extracted codebase knowledge



Q/A assistant with codebase knowledge

6

Challenges

What's next?



Giovanni Rosa, PhD

AI Engineer @Technology Reply

giovannirosa.com

Thank You!



SCAN ME