

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 image shows a screenshot of a ChatGPT interface. On the left, there's a sidebar with various AI-generated writing prompts: "Friendly Profes...", "Brainstorming Bl...", "Cover Letter Crafting", "Brainstorming Blog Po...", "Creative Writing Prom...", "Previous 7 Days", "Building Your Personal...", "Ideating Marketing Ca...", "Designing a Compellin...", and "Refining Your Busines...". The main area shows a conversation between the user ("You") and ChatGPT. You ask, "Can you edit my email to be more friendly, but still professional?" ChatGPT replies, "Certainly! Here's a revised version of the email with a more friendly yet still professional tone:" followed by a sample email. The sample email starts with "Subject: Excited to Move Forward and Connect Further!" and "Hi Judy,". It concludes with "I hope this email finds you well. I wanted to express our genuine excitement ●". At the bottom, there's a message input field with the placeholder "Message ChatGPT..." and a small upward arrow icon.

ChatGPT

Today

Friendly Profes... ⋮

Brainstorming Bl...

Yesterday

Cover Letter Crafting

Brainstorming Blog Po...

Creative Writing Prom...

Previous 7 Days

Building Your Personal...

Ideating Marketing Ca...

Designing a Compellin...

Refining Your Busines...

You

Can you edit my email to be more friendly, but still professional?

ChatGPT

Certainly! Here's a revised version of the email with a more friendly yet still professional tone:

Subject: Excited to Move Forward and Connect Further!

Hi Judy,

I hope this email finds you well. I wanted to express our genuine excitement ●

Message ChatGPT...



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



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

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 access

tools. "We are not new to this field," 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.

6 DAILY NATION TUESDAY, JANUARY 17, 2023

HealthyNation | Tech

How ChatGPT will transform medicine this year

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

ChatGPT: Optimizing Language Models for Dialogue

We've trained a model called ChatGPT - we're trained in a more conventional way. The way to answer questions is to parse them and then to generate responses. It's a very good way to answer questions.



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

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

News in focus



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

6

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 access



solutions, treatments and consultations. And with Kenya's tech ecosystem boomerang, many startups have sprung over the last seven years.

This means it can improve efficiency in the pharmaceutical supply chain by automatically generating purchase orders, invoices and delivery notes.

We are not new to this news. 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.

10

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 comes to what real-world jobs an AI matrix could replace have been brought up. Here experts and tech pundits are saying could be replaced by AI.



Graphic Designers



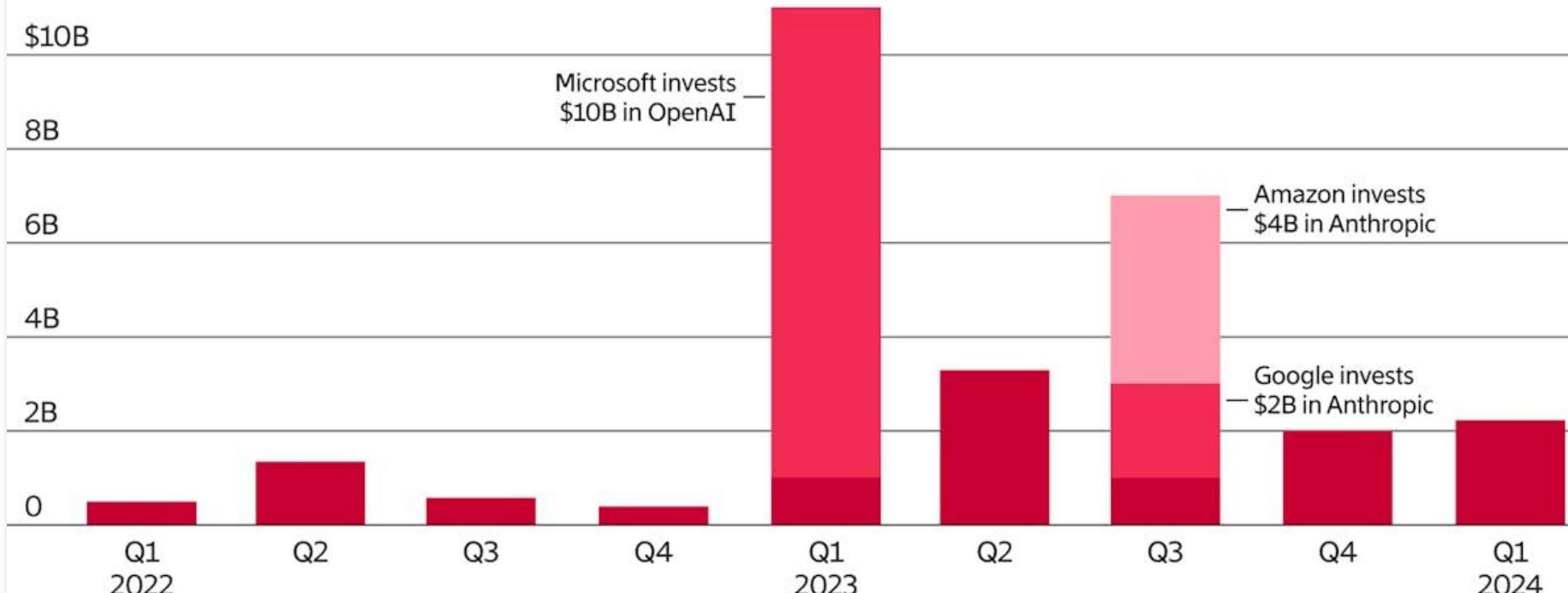
Customer Service Agents

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.

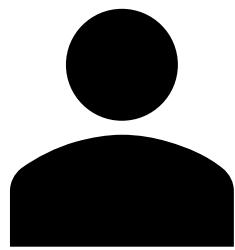
Chances are, you have already

Generating Investment

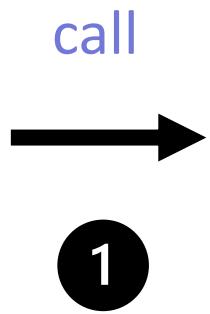
The release of ChatGPT in November 2022 sparked a rush of investment in generative artificial intelligence companies.



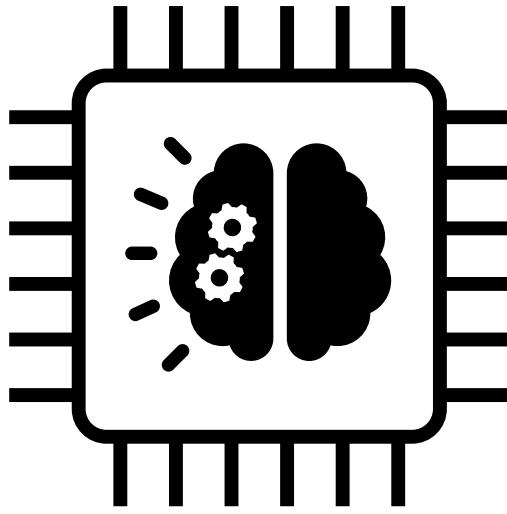
Source: The Information reporting



User



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



**Generative AI
model**

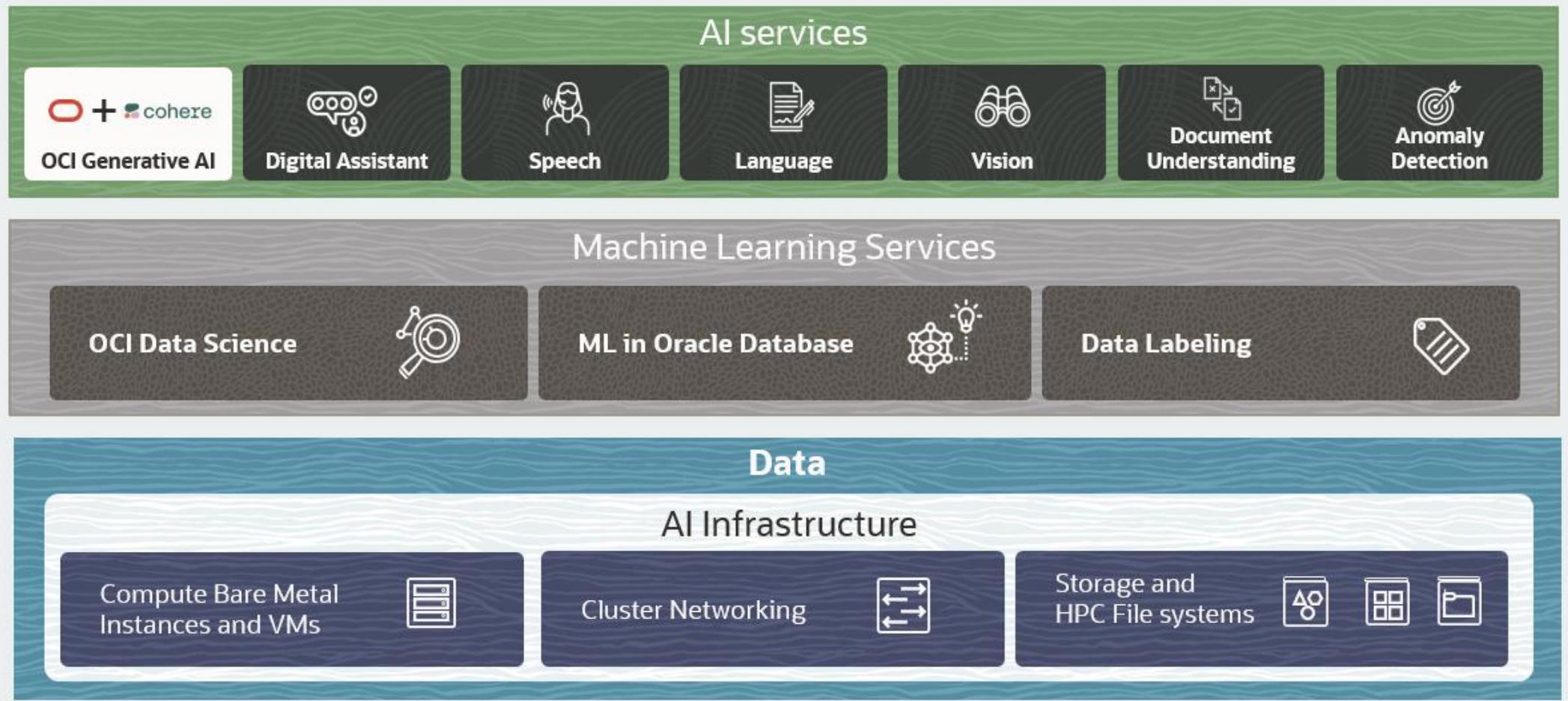
create



2



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

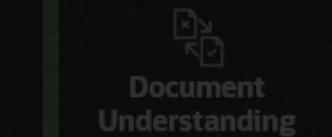
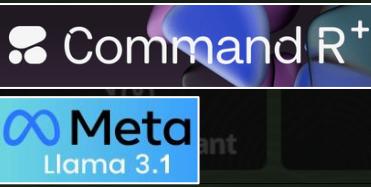


Source: blogs.oracle.com

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



Data

AI Infrastructure

Compute Bare Metal Instances and VMs



Cluster Networking



Storage and HPC File systems

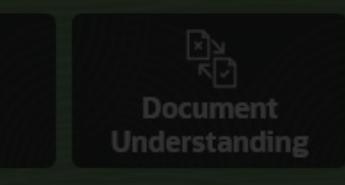
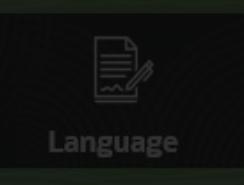
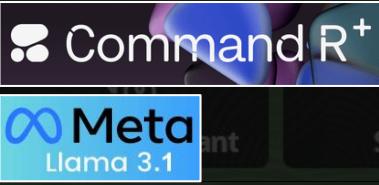


Source: blogs.oracle.com

Oracle AI Services

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

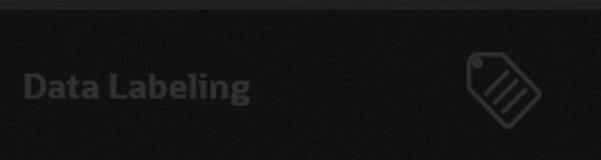
😊 Inference API (pay-per-use)



😊 Dedicated instances

😊 Fine-tune models

Machine Learning Services



Compute Bare Metal
Instances and VMs

😊 Local SLM - no GPU!

😊 Vector store

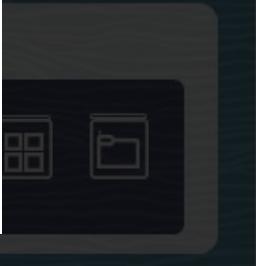
😊 AutoML

AI Infrastructure

Cluster Networking



Storage HPC

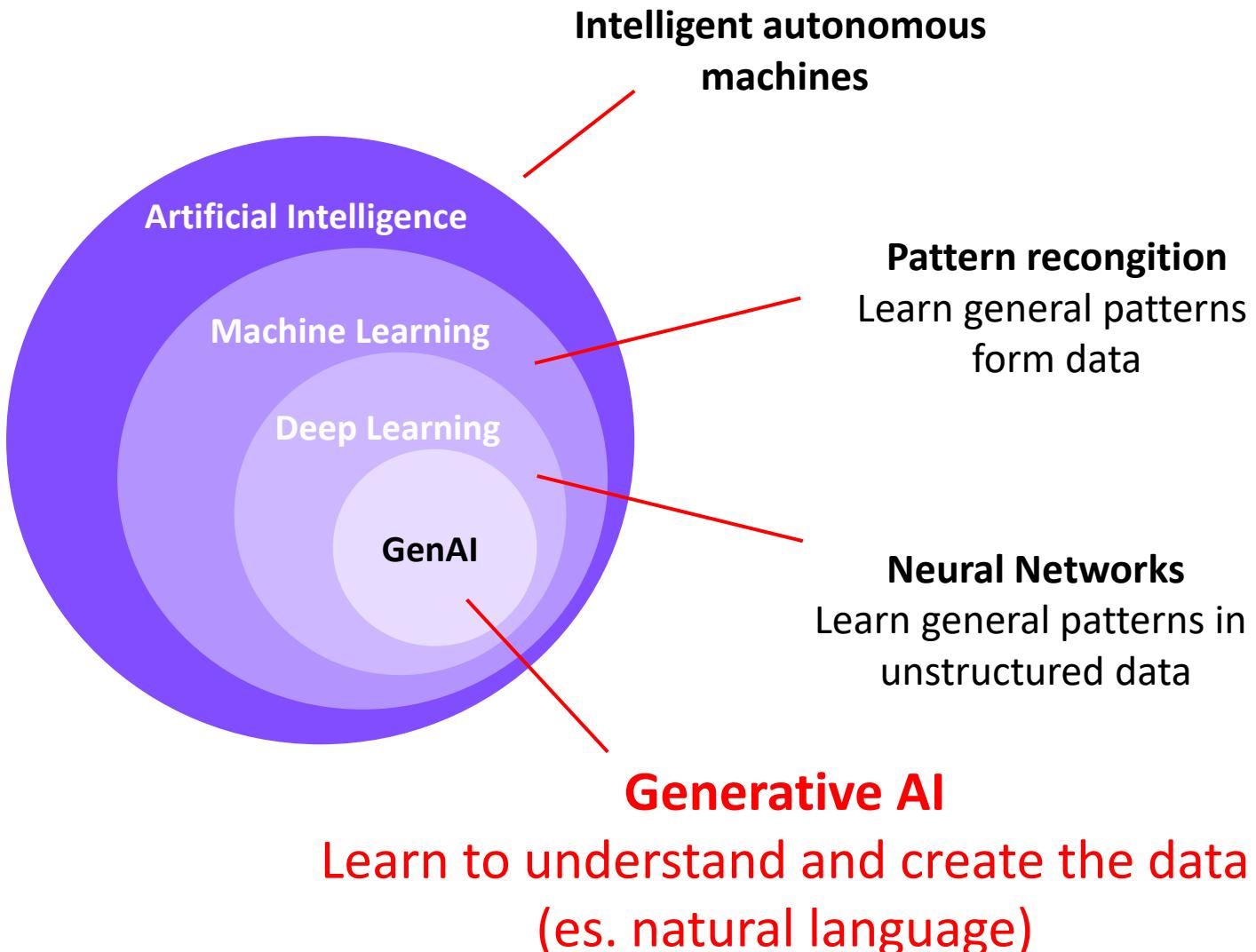


Source: blogs.oracle.com

Oracle AI Services

What is Generative AI

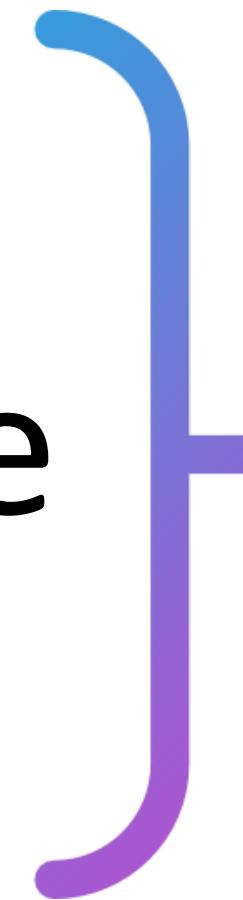




Large

Language

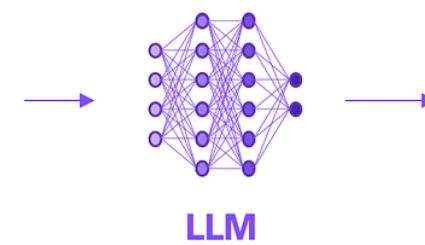
Model



LLM

A trained language model can generate coherent...

A trained language model can
Input



Word	Probability
speak	0.065
generate	0.072
politics	0.001
...	...
walk	0.003

Output at step 1

2

Word	Probability
ability	0.002
text	0.084
coherent	0.085
...	...
ideas	0.041

Output at step 2

A trained language model can generate...

1

Source: medium.com

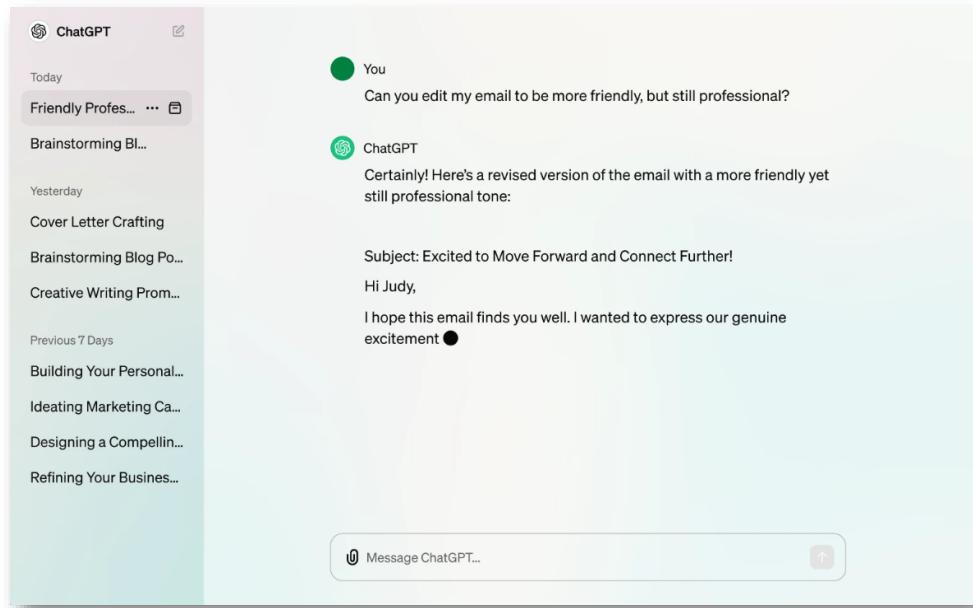
How does a language model work?

G

P

T

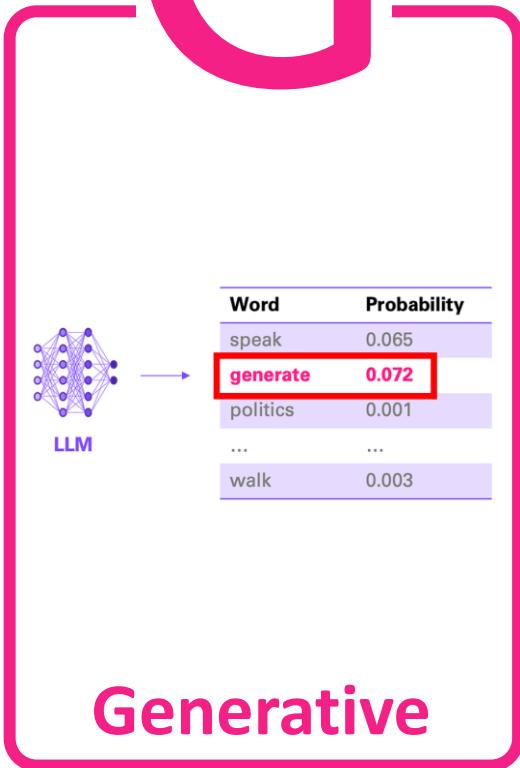
3.5

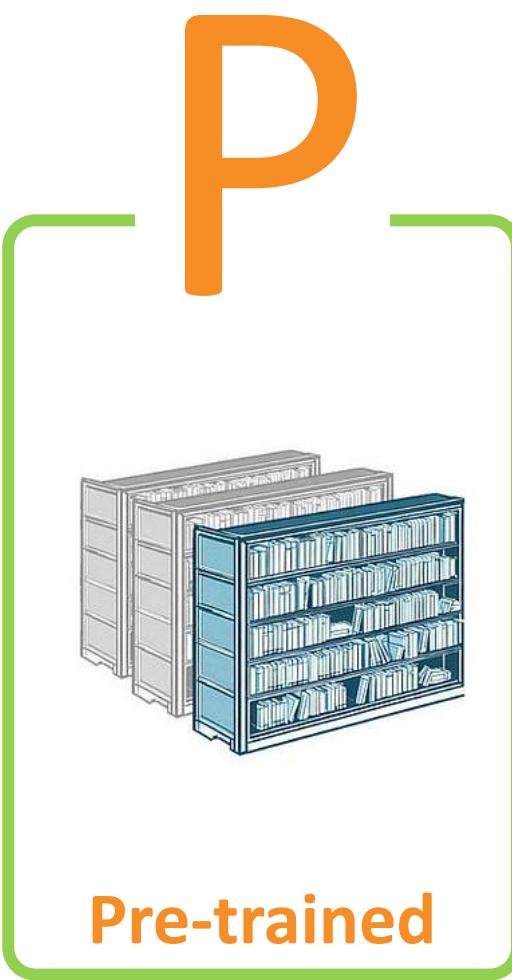
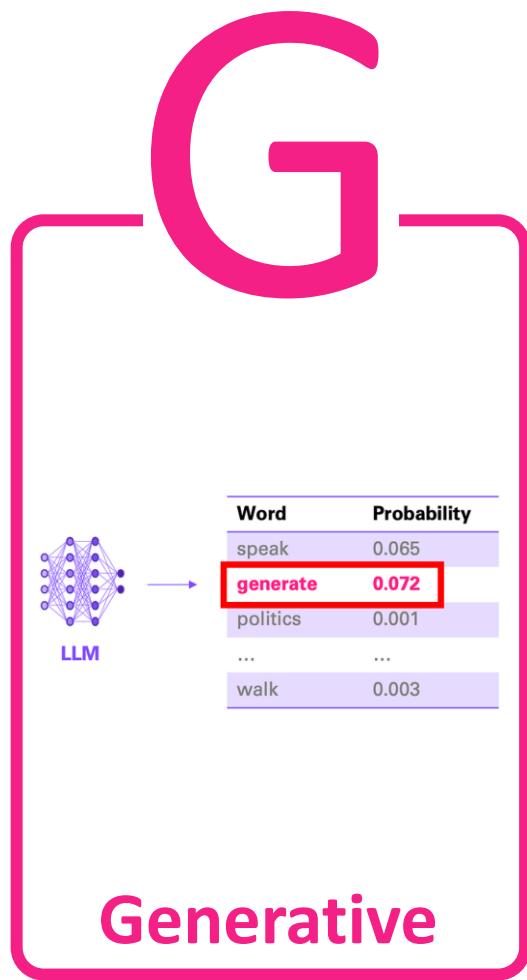


G

P

T

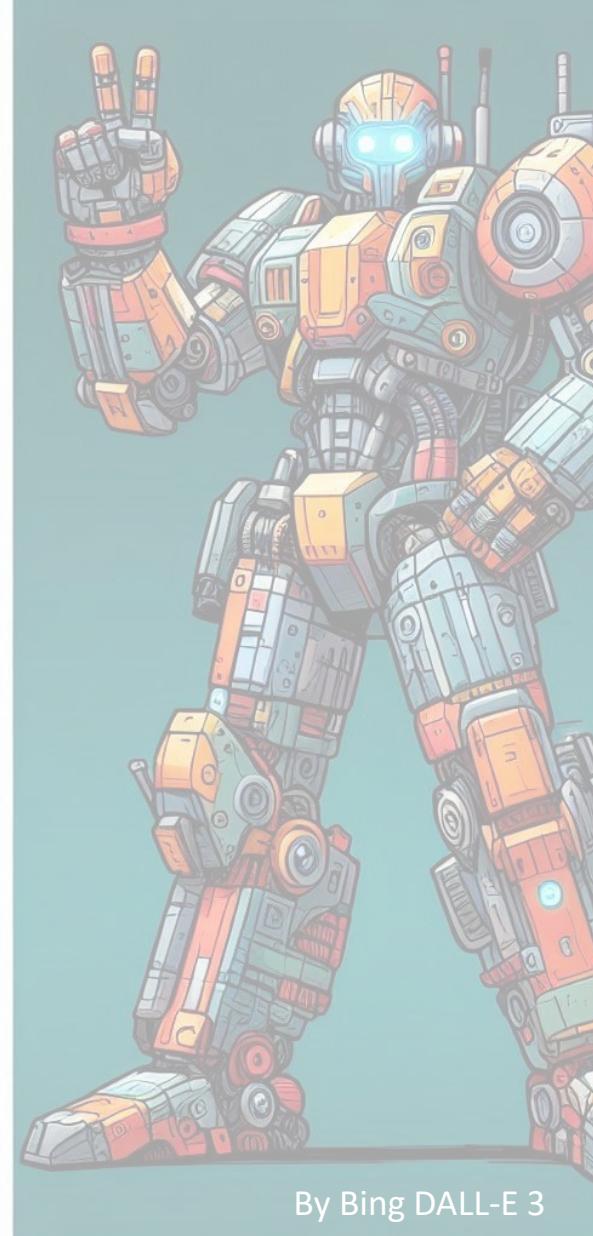
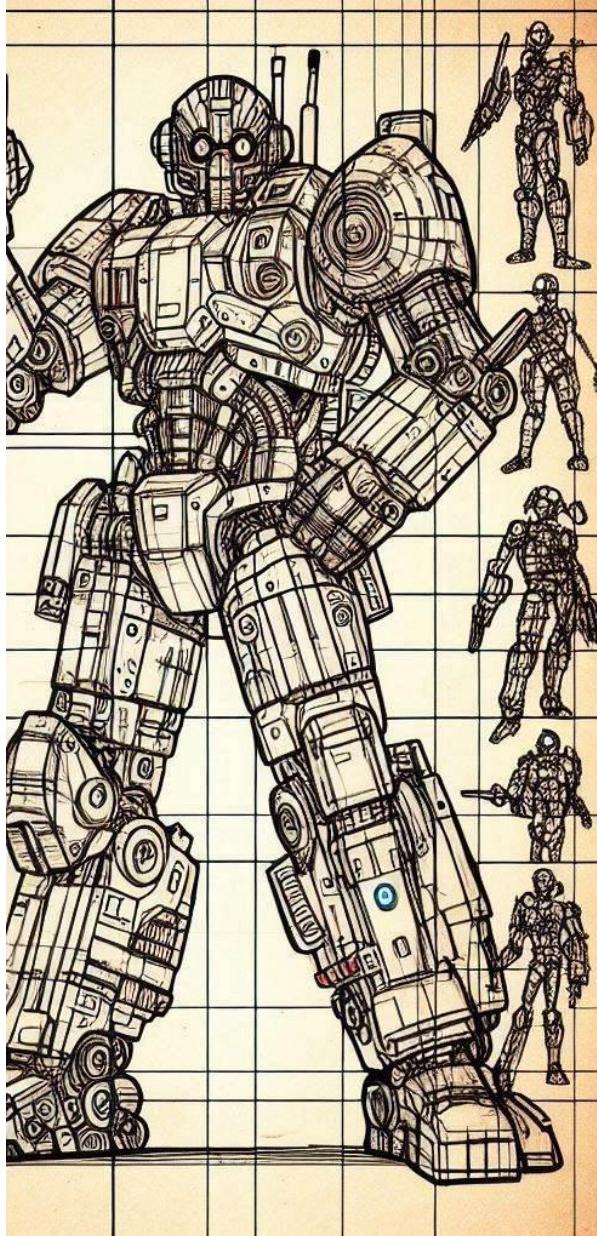




T

Pre-training

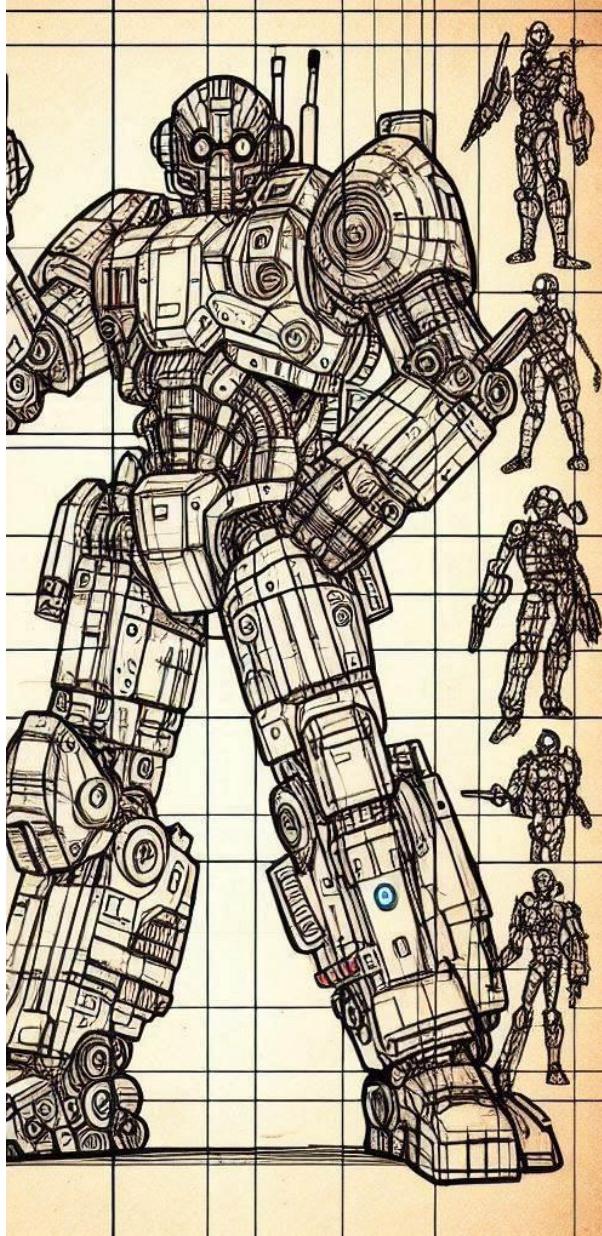
Learn
language structure
and knowledge



How to train a «Large» Language Model?

Pre-training

Learn
language structure
and knowledge



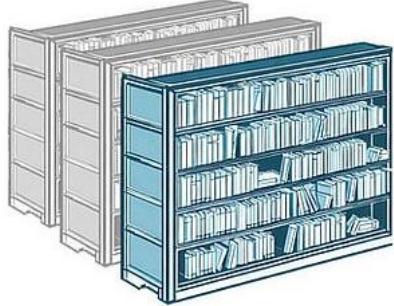
Instruction Fine-tuning

Learn
how to chat

How to train a «Large» Language Model?

GPT-4

Est. train size



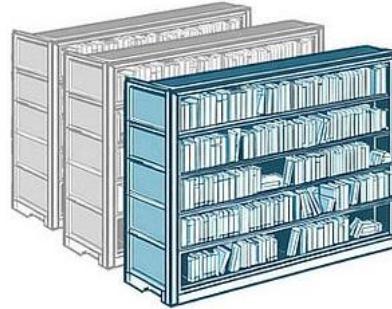
650 km

Line of library shelves

Pre-training Large Language Models

GPT-4

Est. train size

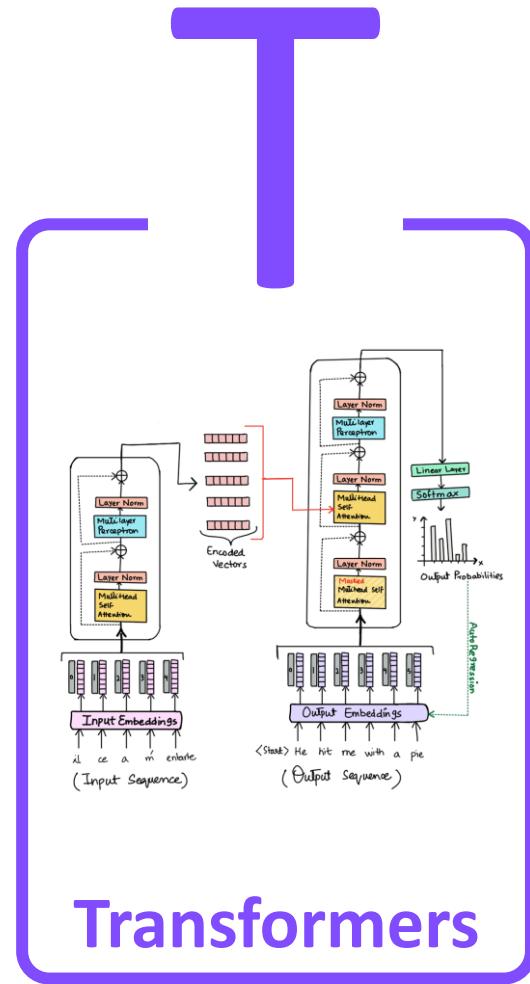
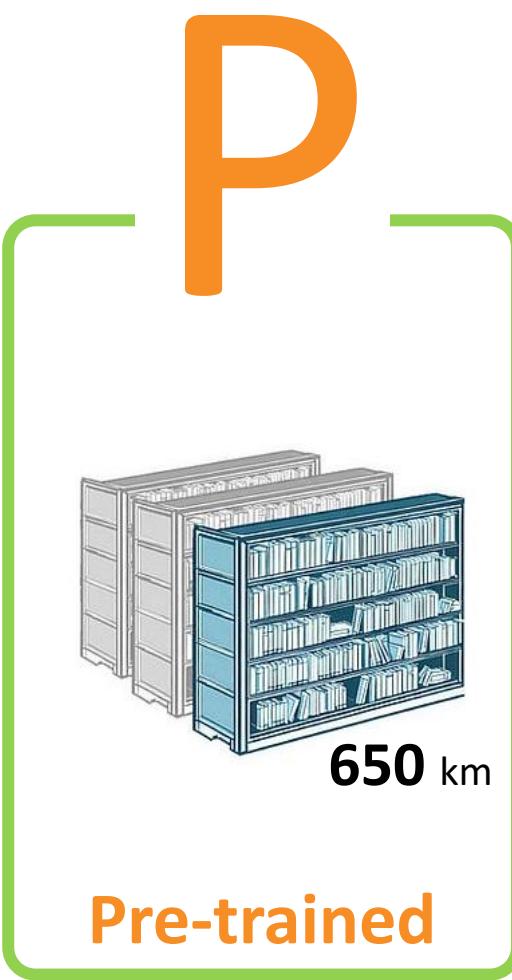
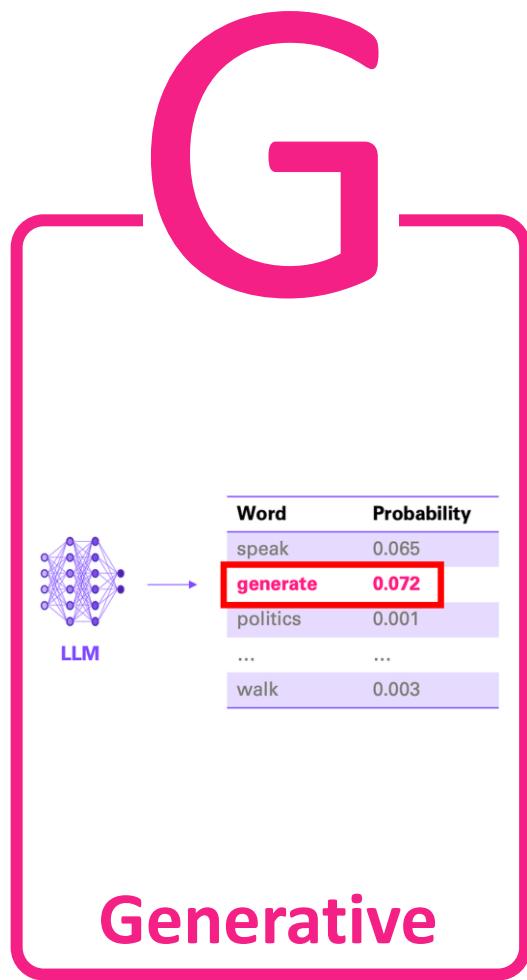


650 km

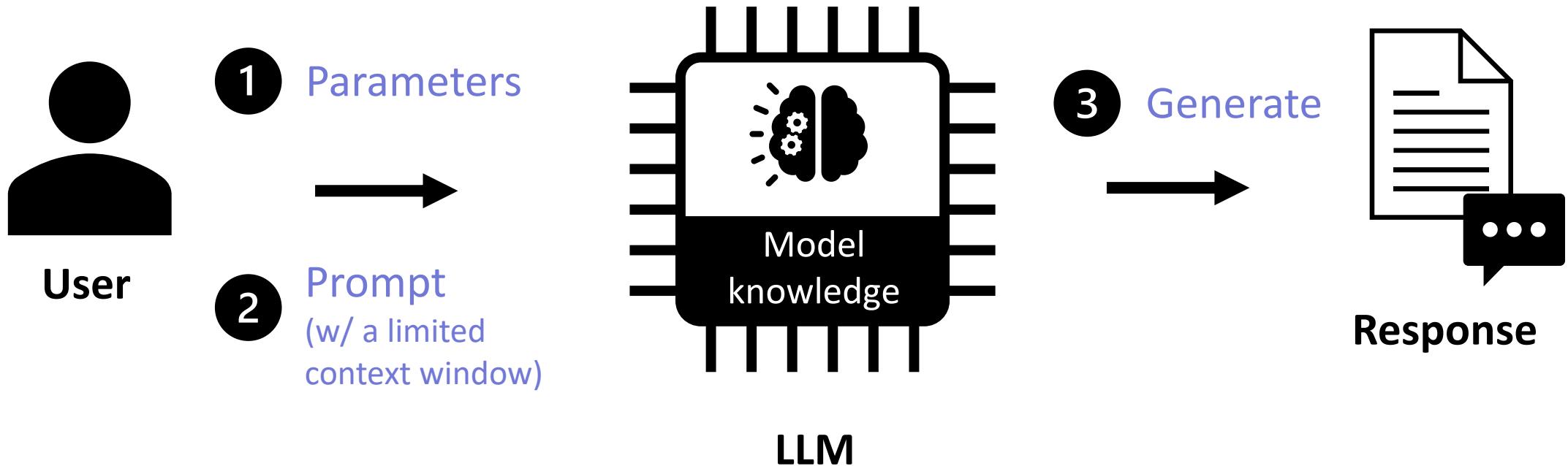
Line of library shelves



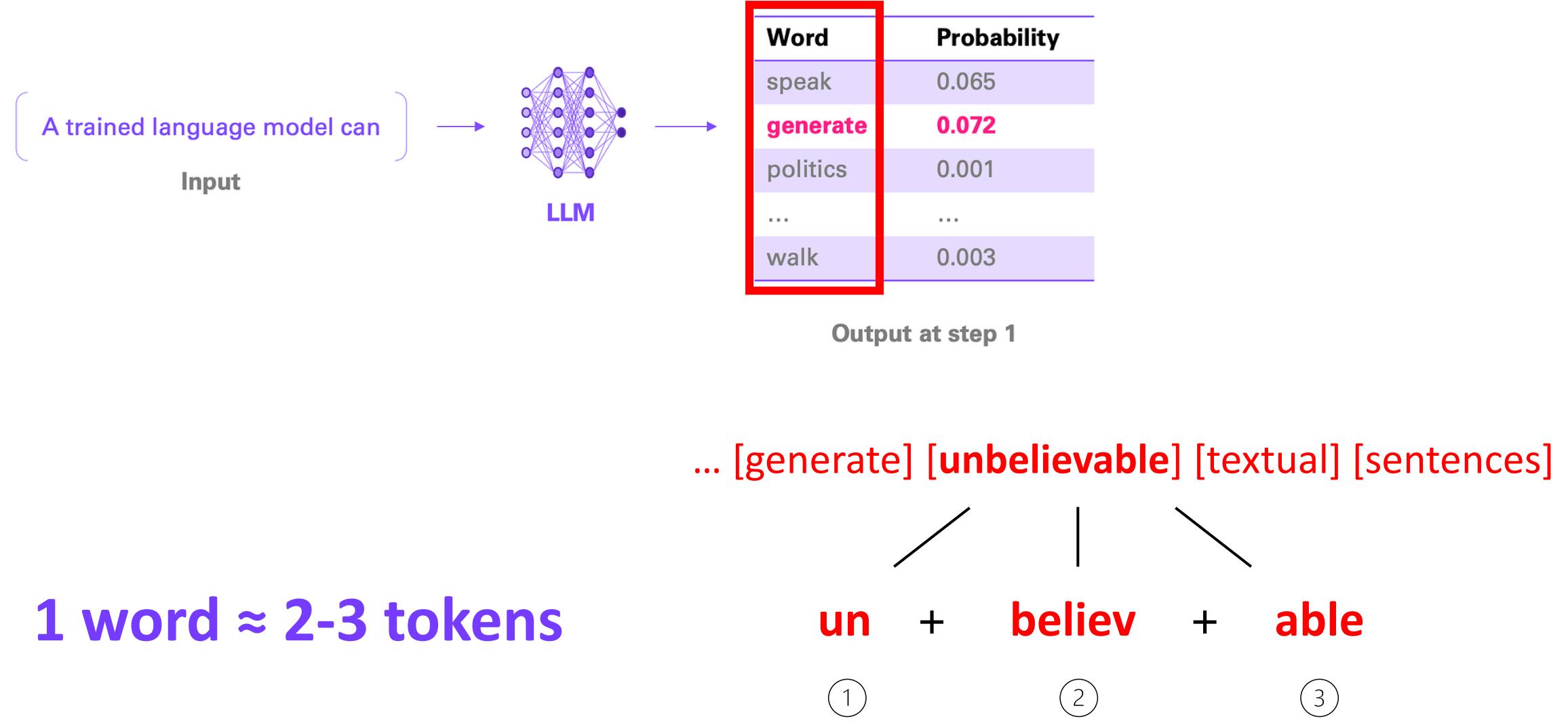
Pre-training Large Language Models



Sources: Sudipto Baul - medium.com



How to instruct LLMs ?



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

Type a message...

Submit

Clear chat

Parameters

Maximum output tokens 600

Preamble override

Provide concise responses

Temperature 0.25

Top p 0.75

Top k 0

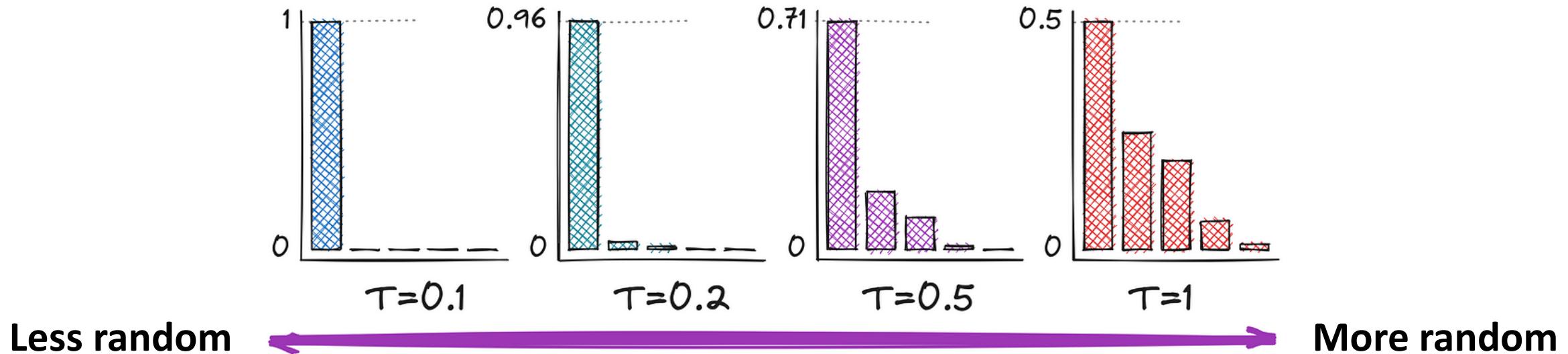
Frequency penalty 0

Presence penalty 0

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 (i)



600

Preamble override (i)

Provide a concise response

Temperature (i)



1

Temperature = 1

Top P (i)



0

Frequency penalty (i)



0

Presence penalty (i)



0



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 ⓘ **Temperature = 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

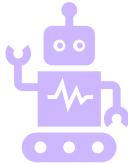
Il servizio clienti non funziona. Ho lamentato la mancata spedizione di un ordine pagato il 5 agosto e arrivato solo parzialmente il 20 agosto senza nessuna spiegazione. Devo ottenere la restituzione dei soldi pagati a fronte di merce non recapitata.



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.promptingguide.ai



By Bing DALL-E 3

Section delimiters

Role

Task description

Desired Output

SYSTEM

You are an advanced NLP classification model.

Task

Your task is to classify the sentiment of the input movie reviews as either 'positive' or 'negative'.

Output format

Return as a response ***ONLY ONE*** of the following strings:
- "Positive", for positive sentiment;
- "Negative", for negative sentiment.

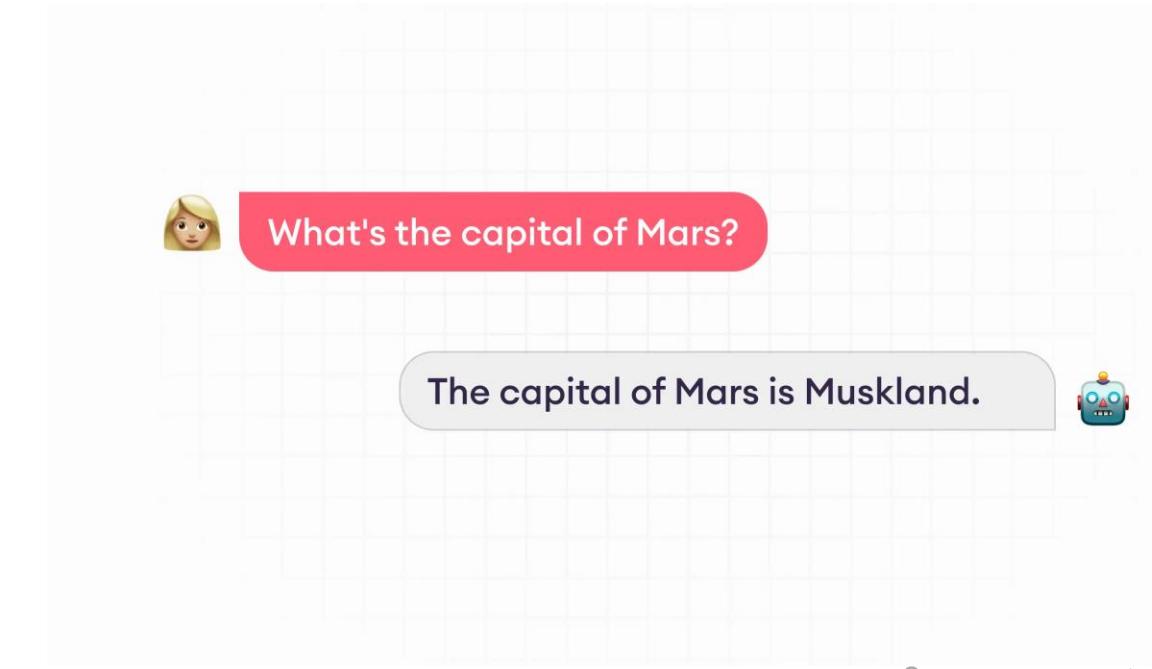
Gives more relevance

How to write a good prompt?

Beware of Hallucinations!

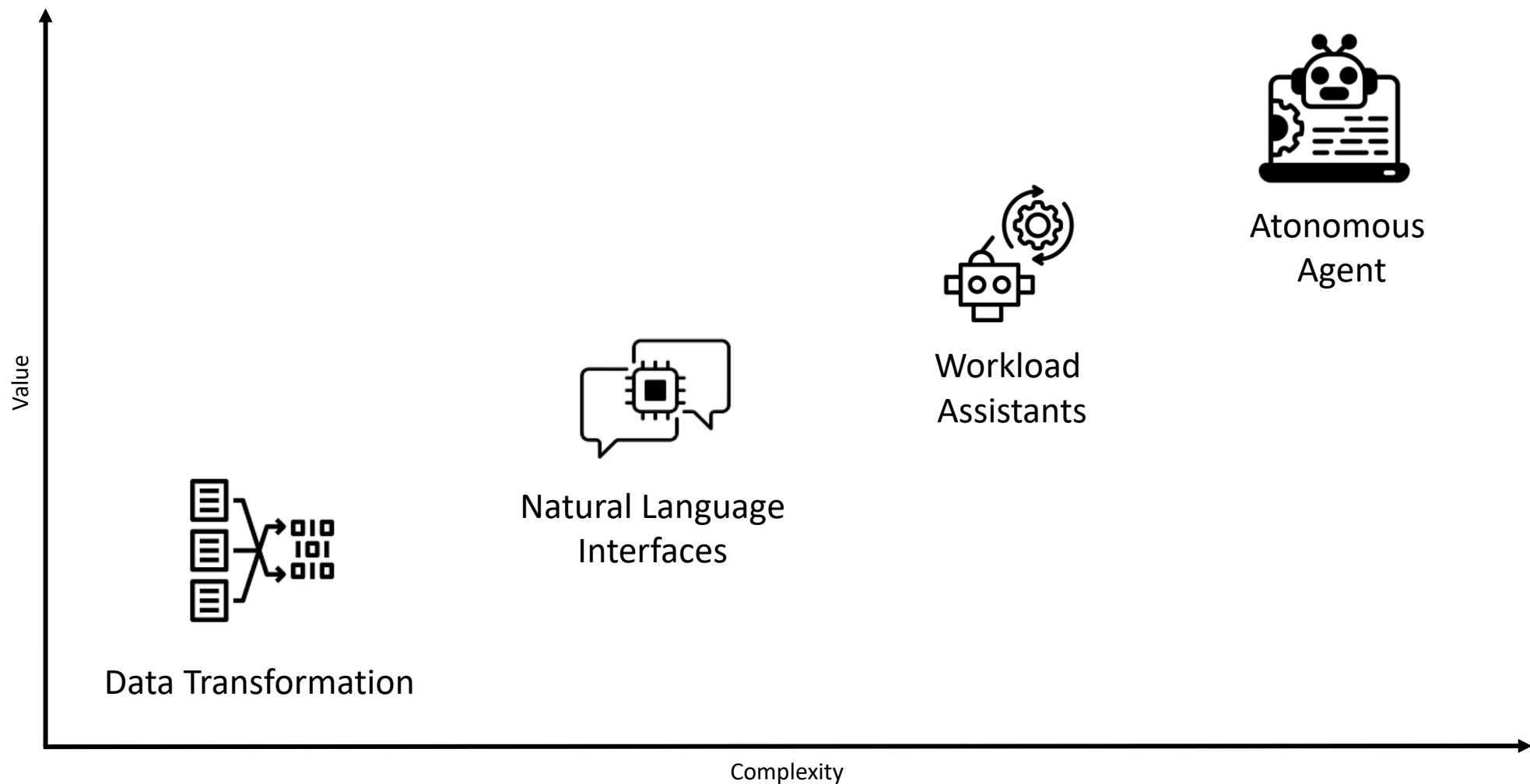


- ✖ Ambiguous prompt
- ✖ Anomalies in training data
- ✖ Outdated model knowledge



Source: superannotate.com

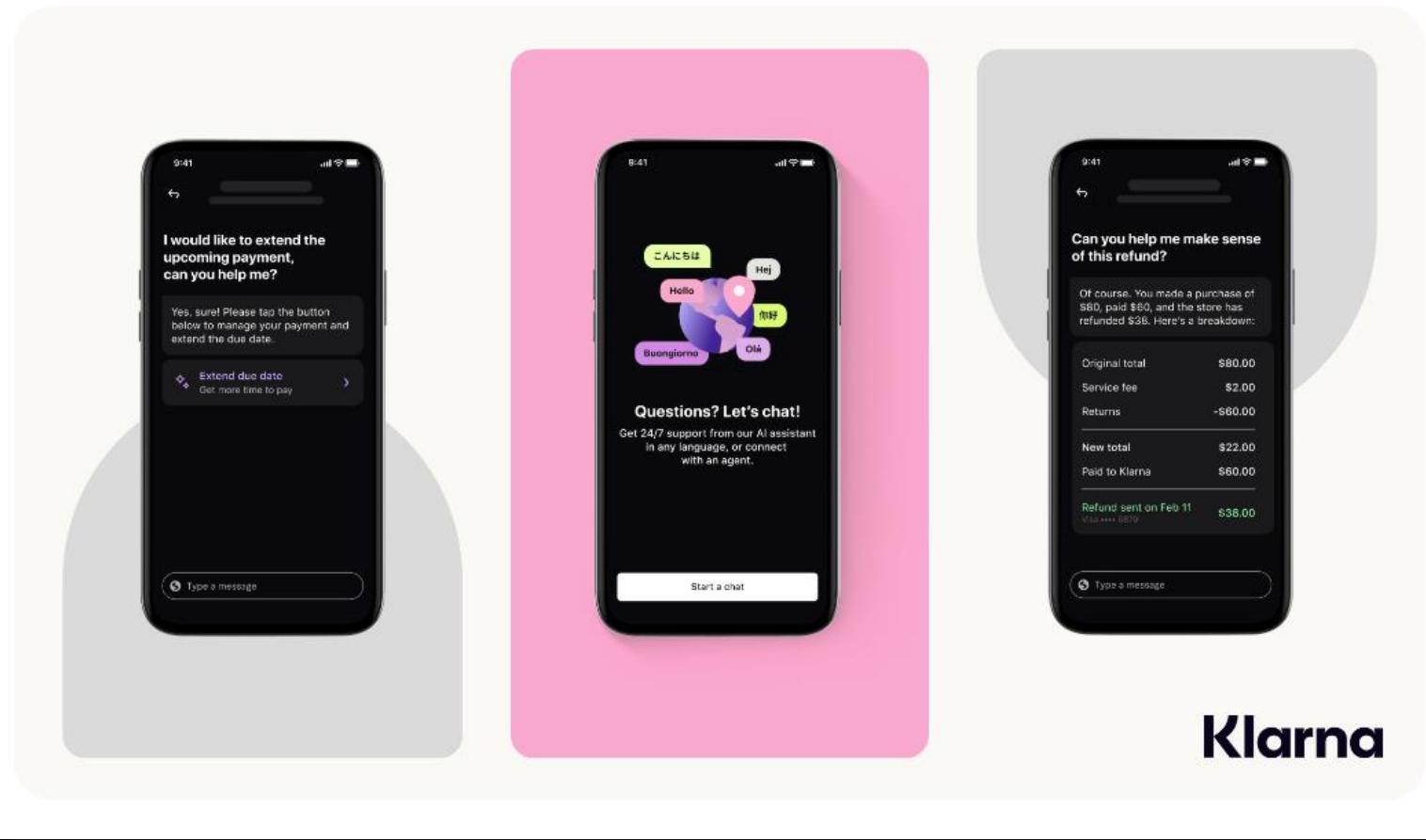
GenAI Use Cases



GenAI use cases

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

February 27, 2024



```
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
```

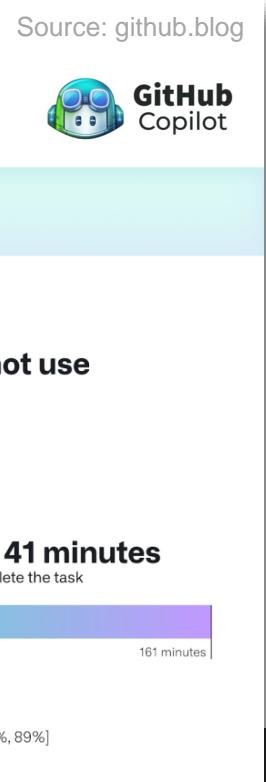
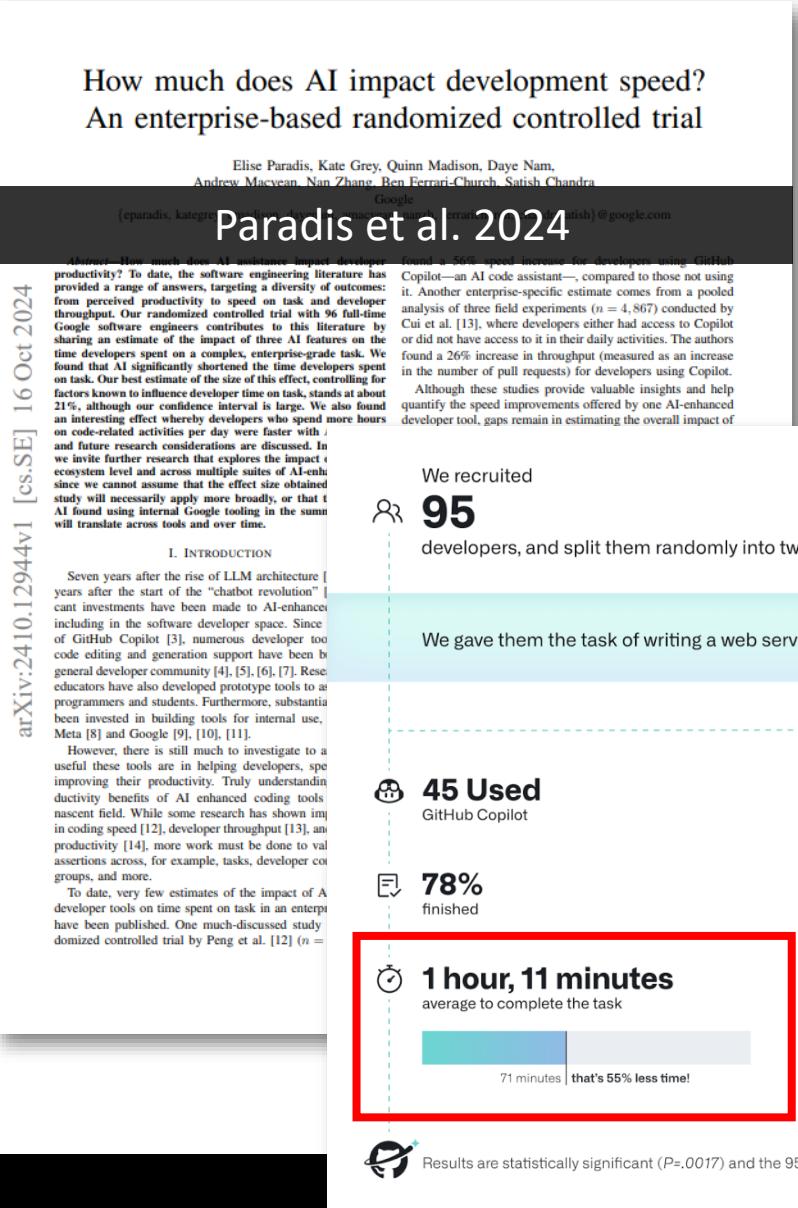


codeium

ORACLE®
Code Assistant



GitHub
Copilot



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 Management University, Singapore

KAILONG WANG, Huazhong University of Science and Technology, China

LI LI, Beihang University, China

XIAOJIU LUO, The Hong Kong Polytechnic University, China

* 2024

Hou et al. 2023-

Large Language Models for Test-Free Fault Localization

Aidan Z.H. Yang
aidan@cmu.edu

Carnegie Mellon University
Pittsburgh, United States
Yang et al. 2024-

Ruben Martins
rubenm@cmu.edu

Carnegie Mellon University
Pittsburgh, United States

Claire Le Goues
clegoues@cs.cmu.edu

Carnegie Mellon University
Pittsburgh, United States
Ruben Martins
rubenm@cmu.edu

Carnegie Mellon University
Pittsburgh, United States

domains, including Software Engineering (SE) and various SE tasks. Nevertheless, a significant portion of the literature on LLMs for SE is still in its early stages, with limited scope and outcomes. We select and analyze 12 studies from 2020 to 2023, our key research questions (RQs), and their contributions. RQ1 explores the potential of LLMs for SE tasks, characterizing their distinctive features, such as pre-training, fine-tuning, preprocessing, and application. RQ2 investigates the challenges and opportunities in applying LLMs to SE. RQ3 examines the practical contributions of LLMs to SE, including their impact on software development, testing, and maintenance. RQ4 examines the limitations and challenges of LLMs in SE. Finally, RQ5 investigates the future research directions and applications of LLMs in SE. Finally, RQ6 examines the practical contributions of LLMs to SE, including their impact on software development, testing, and maintenance. RQ7 examines the limitations and challenges of LLMs in SE. Finally, RQ8 examines the future research directions and applications of LLMs in SE.

ABSTRACT

Fault Localization (FL) aims to automatically localize buggy lines of code, a key first step in many manual and automatic debugging tasks. Previous FL techniques assume the provision of input test cases 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. We 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 10 billion parameters on small, manually curated corpora of buggy 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. In *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. © 2024 copyright held by the author(s). ACM ISBN 978-1-4503-6607-0/24/04 \$15.00.
<https://doi.org/10.1145/3597503.3623342>

Using an LLM to Help With Code Understanding

Daye Nam
Carnegie Mellon University
U.S.A.
dayen@cs.cmu.edu

Andrew Macvean
Google Inc.
U.S.A.
andrew.macvean@google.com



Vincent Hellendoorn
Carnegie Mellon University
U.S.A.
hellend@andrew.cmu.edu

Bogdan Vasilescu
Carnegie Mellon University
U.S.A.
vasilescu@cs.cmu.edu

Brad Myers
Carnegie Mellon University
U.S.A.
bam@cs.cmu.edu

ABSTRACT

Understanding code is challenging, especially when working in modern 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 that is geared toward 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.

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 engines [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 tool via a local database. The tool provides a simple interface for users to interact with the LLM and receive information related to their current codebase.

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

Code generation

Code understanding

Requirements engineering

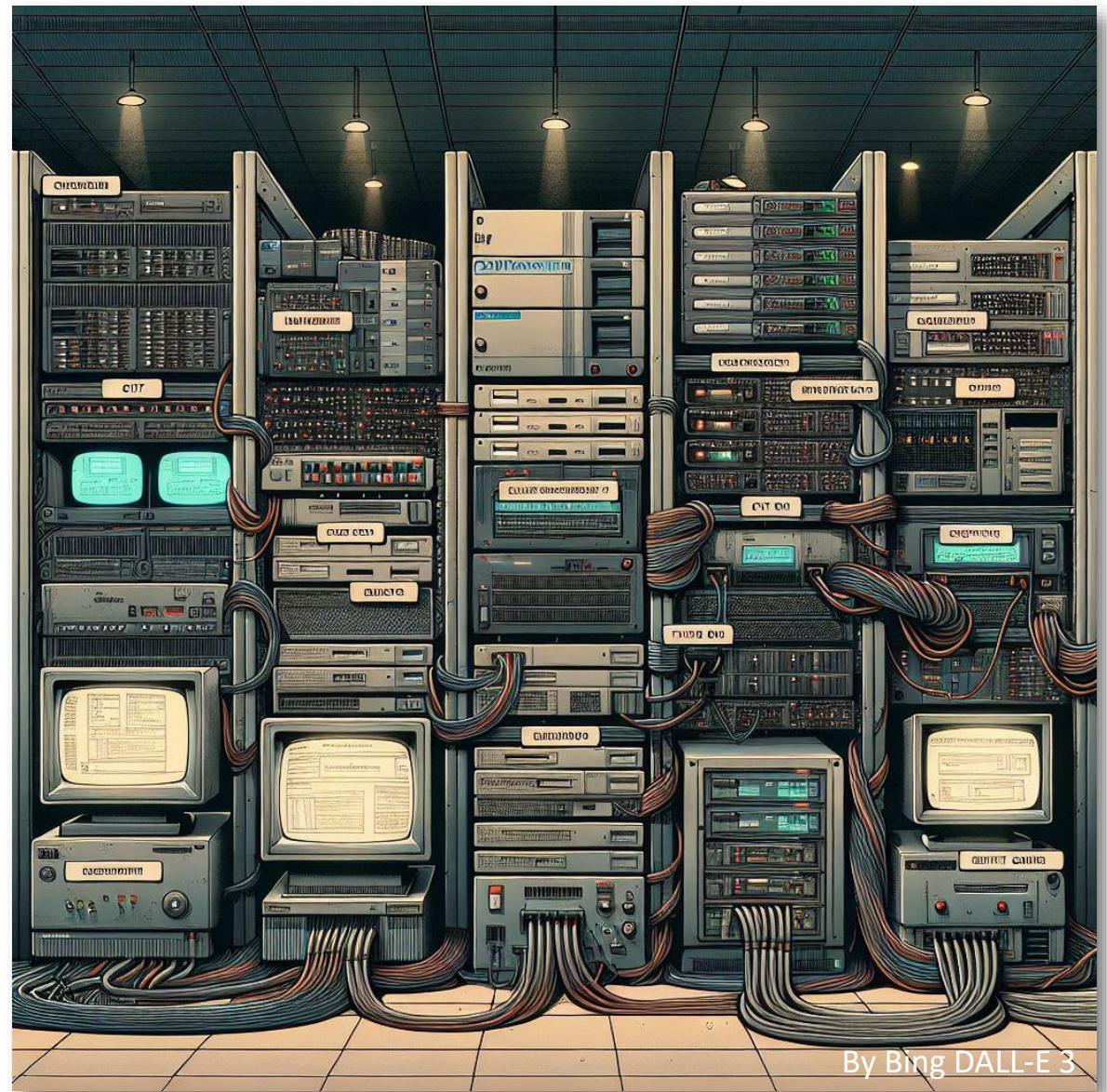
Bug-fixing

... and more

Legacy Systems

Old system or application
that is still in use

Business-critical for the
company



Legacy Systems



**Higher
maintenance costs**

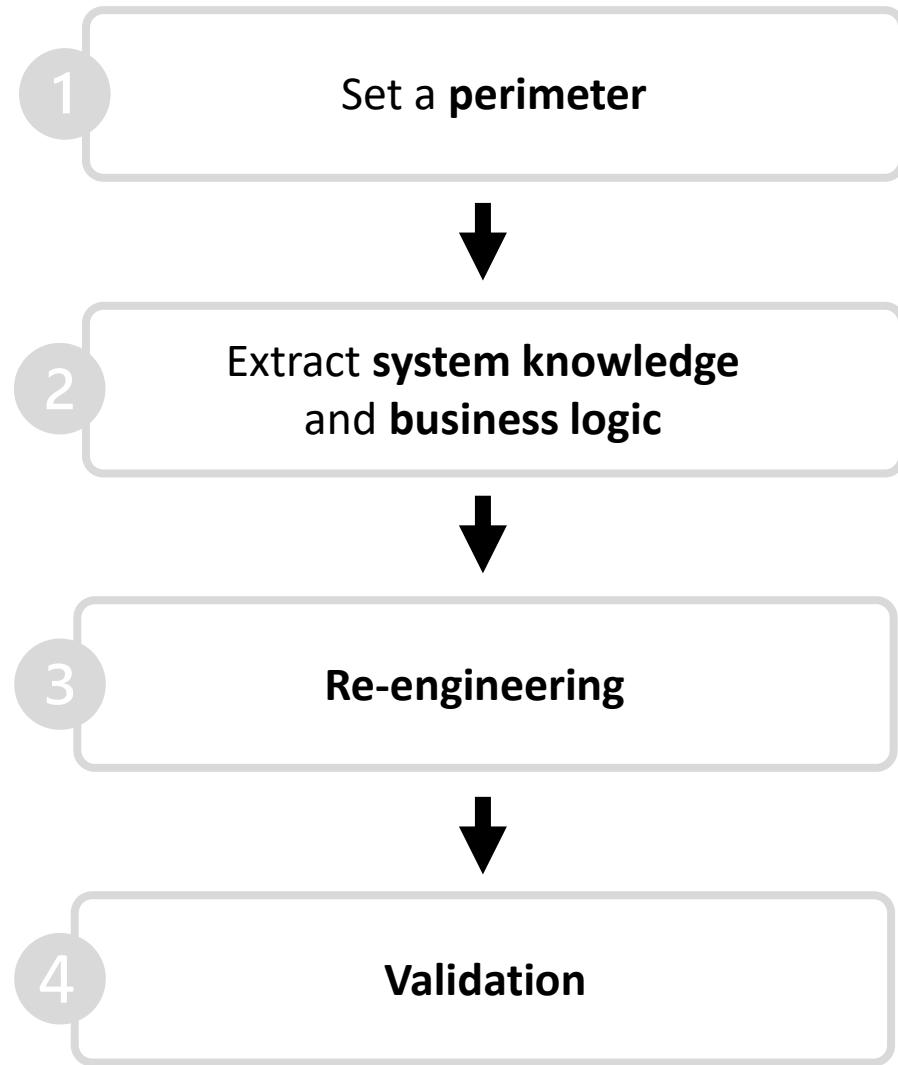


Difficult to extend to new
business requirements

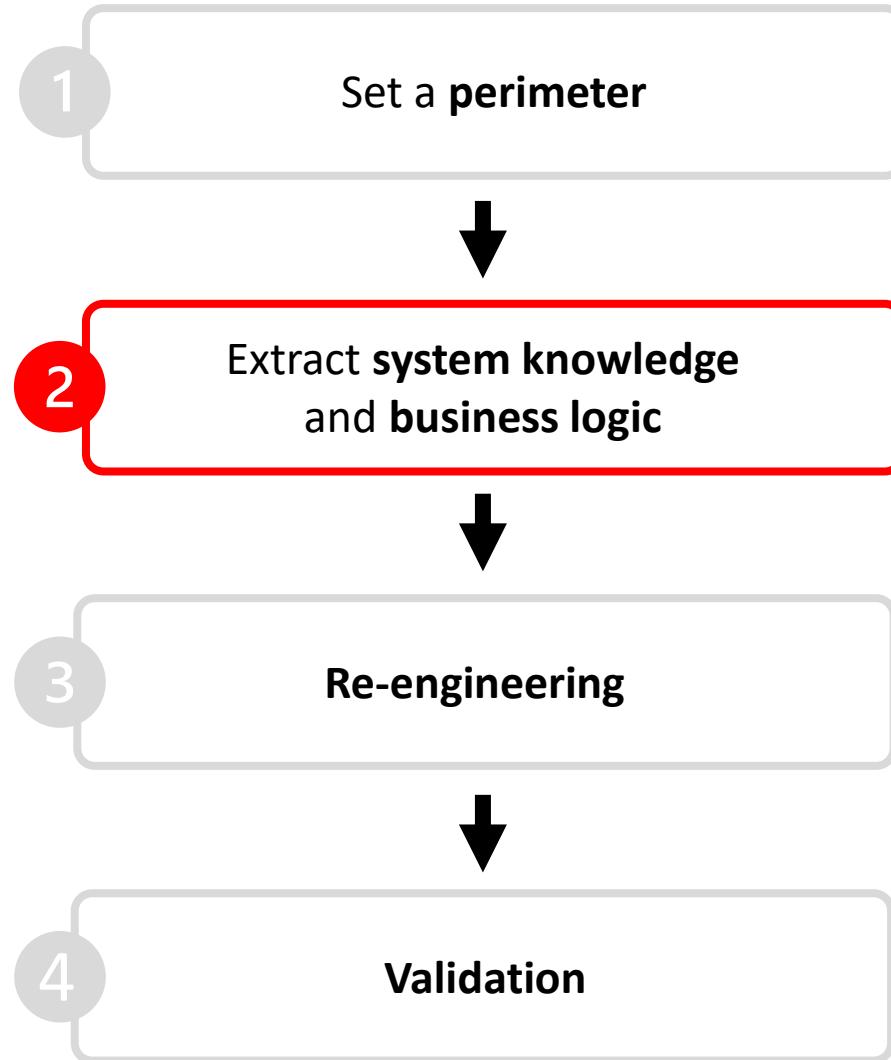


Proneness to **quality**
and **reliability** issues

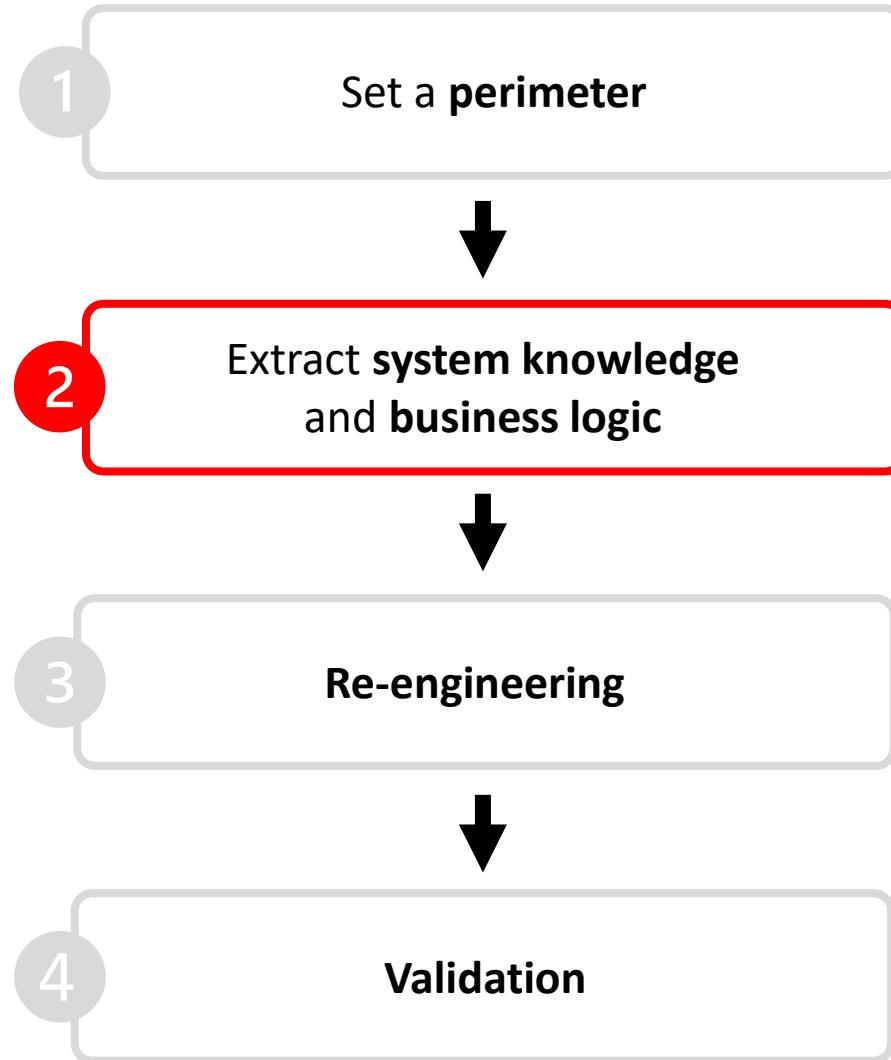
Driver for legacy system modernization



Modernizing Legacy Code



Extracting the encapsulated
system and business knowledge
is hard



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 Liu², Gelei Deng², Yang Liu², Zhenyu Chen¹

¹State Key Laboratory of Novel Software Technology, Nanjing University, Nanjing, China

²College of Computer Science and Technology, Nanyang Technological University, Singapore, 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@e.ntu.edu.sg, gelei.deng@ntu.edu.sg, yangliu@ntu.edu.sg, zychen@nju.edu.cn

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,

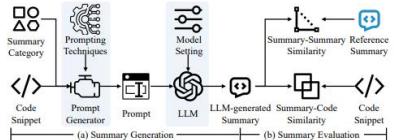


Fig. 1: General workflow of LLM-based code summarization and its effectiveness evaluation

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.

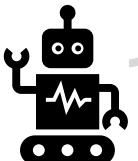
Explain in a few words the purpose of the input code snippet for a non-technical person.

```
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 }
```

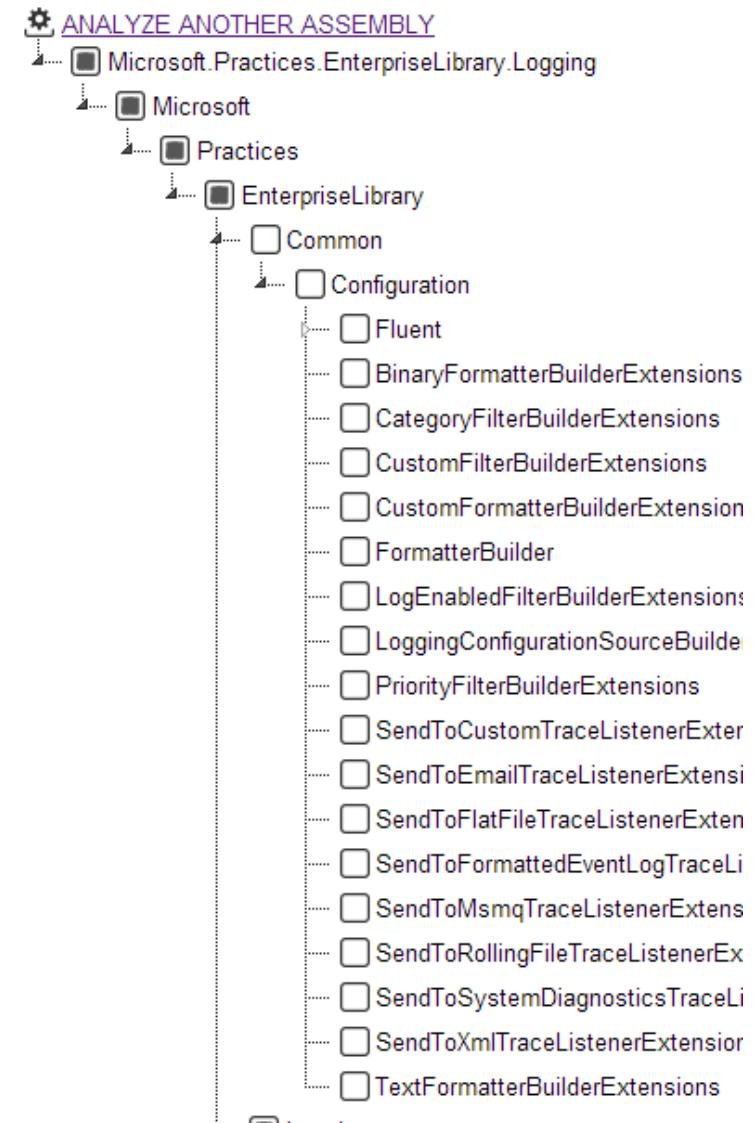


This code is designed to check the validity of a number using the **Luhn algorithm**, which is a simple method often used to verify credit card numbers.

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



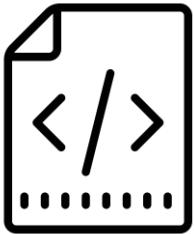
Code Summarization: Example



Source: gmamaladze.wordpress.com

How to summarize complex codebases ?

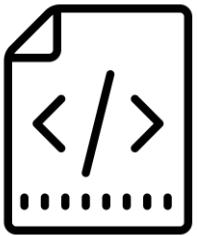
1



Select input
sources

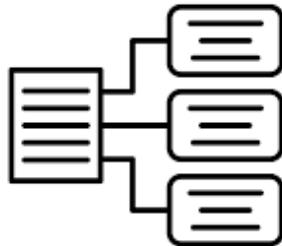
Summarizing a complex codebase

1



Select input
sources

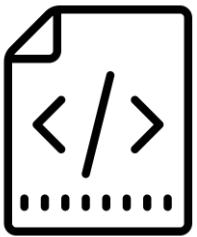
2



Extract sources
structure and
dependencies

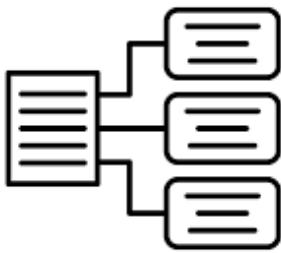
Summarizing a complex codebase

1



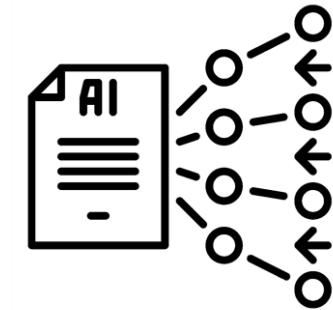
Select input
sources

2



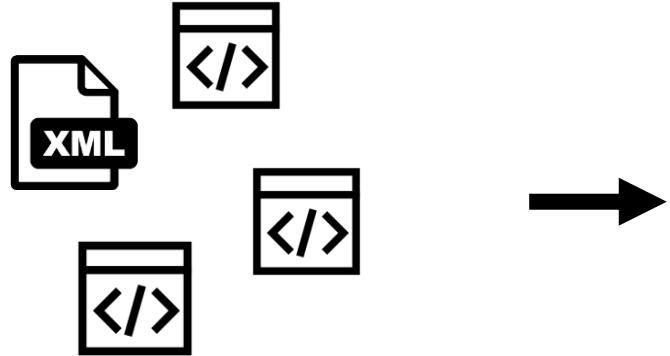
Extract sources
structure and
dependencies

3



Generate natural
language summary

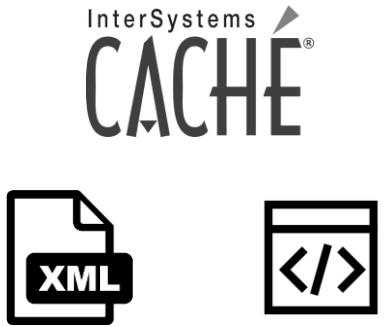
Summarizing a complex codebase



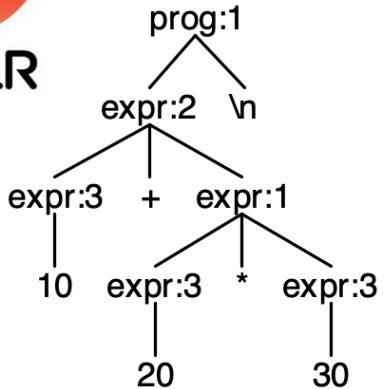
Codebase
subset

1

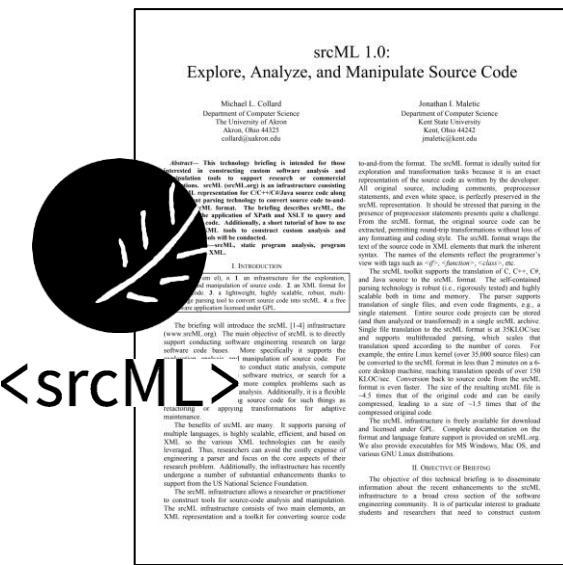
Select input sources



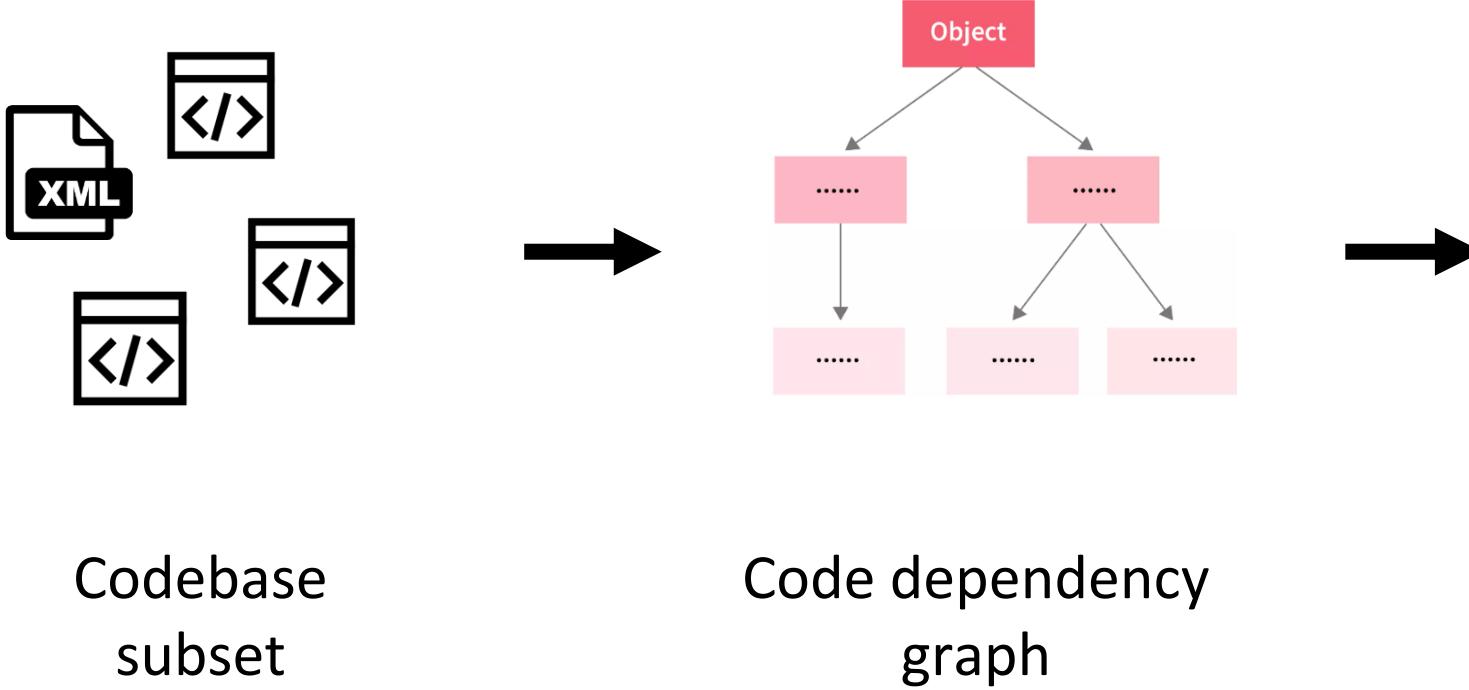
Language-native exporting tools



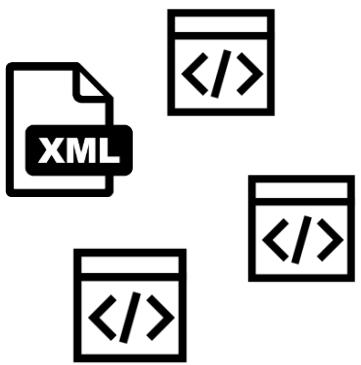
Custom code parsers



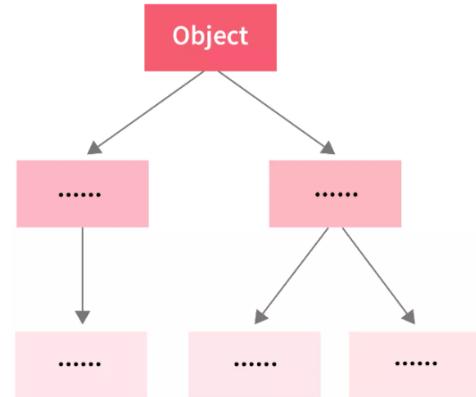
Existing code analysis tools



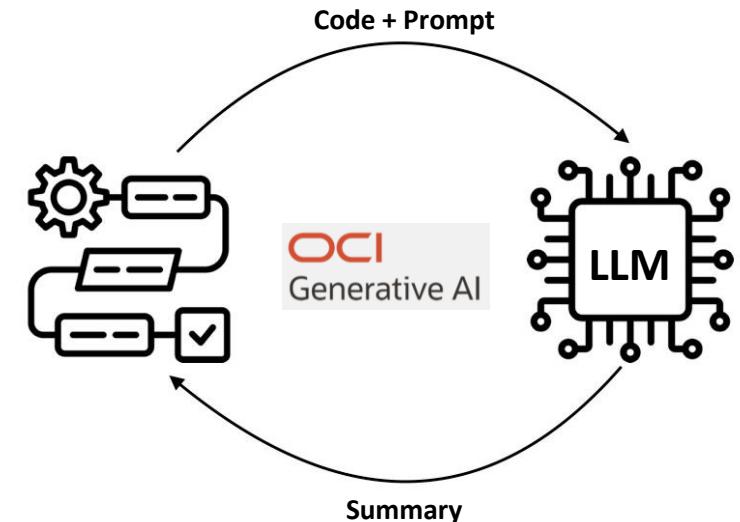
2 Build codebase structure and dependencies



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    // Step 2: Check if patient exists
29    Set patient = #&class(Hospital.Patient).%OpenId(patientID)
30    If 'patient
31    | Return $$$ERROR($$$GeneralError, "Patient not found.")
32
33
34    // Step 3: Assign initial values
35    Set ..PatientID = patientID
36    Set ..AssignedDoctorID = doctorID
37    Set ..AdmissionDate = %ZDATE($HOROLOG,8)
38    Set ..Procedures = #&class (%ListOfTypeDataTypes).%New()
39    Set ..AdmissionStatus = "Pending"
40
41    // Step 4: Request insurance approval
42    Set status = ..RequestInsuranceApproval()
43    If $$$ISERR(status) {
44      Return status
45    }
46
47    // Step 5: Assign a room
48    Set status = ..AssignRoom()
49    If $$$ISERR(status) {
50      Return status
51    }
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, dimissions, medical treatments, allergies, etc.

(Written in Caché ObjecScript)



Example Scenario

1

Setup the LLM inference

Setup authentication

```
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
17 # Initialize the LLM
18 llm = OCIGenAI(
19     model="cohere.command-r-16K", # e.g., "cohere.command-r-plus", "meta.llama-3.1-70b-instruct"
20     service_endpoint="https://inference.generativeai.eu-frankfurt-1.oci.oraclecloud.com", # EU service endpoint
21     compartment_id=<YOUR.OCI.COMPARTMENT.ID>, # your compartment ID from OCI
22     temperature=0.7
23 )
24
25 # Run inference
26 response = llm.chat(messages)
27 print(response)
28
29 [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.
```

Prompts

LLM parameters



```
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 """
```

1

Extract high-level description of class functionalities

2

Extract implementation logic from methods

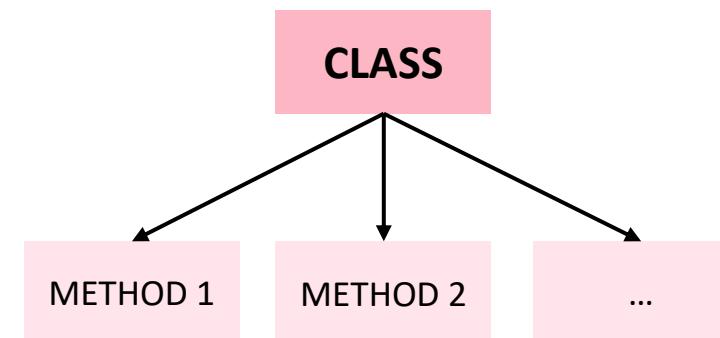
```
1 system_prompt_method = """
2 Impersonate a Caché ObjectScript software engineer.
3
4 # **Steps**
5 Step by step, describe in brief what the provided method implementation does in such a way a person without domain knowledge
6 of the entire codebase can understand and reimplement it in another programming 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 """
```

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 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 = %DATE($HOROLOG,8)
39        Set ..Procedures = #&class(XListOfDataTypes).%New()
40        Set ..AdmissionStatus = "Pending"
41
42
43        // Step 4: Request insurance approval
44        Set status = ..RequestInsuranceApproval()
45        If $$$ISERR(status)
46        | Return status
47
48
49        // Step 5: Assign a room
50        Set status = ..AssignRoom()
51        If $$$ISERR(status)
52        | Return status
53
54
55        // Step 6: Update admission status
56        Set ..AdmissionStatus = "Active"
57
58
59        // Step 7: Save the admission record
60        Set status = ..%Save()
61        If $$$ISERR(status)
62        | Return status
63
64
65        Write "Admission initiated successfully. Admission ID: ", ..AdmissionID, !
66        Quit $$OK
67    }
```

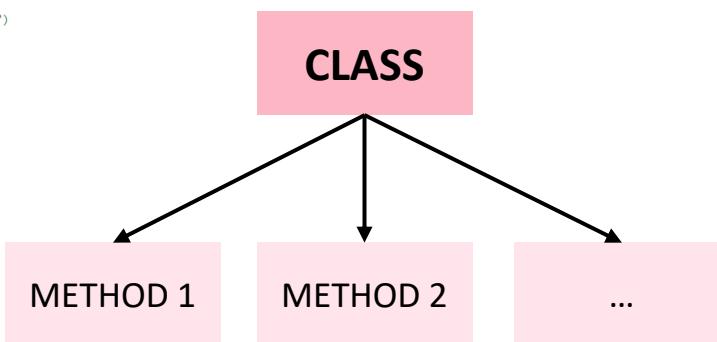
Extract class
features



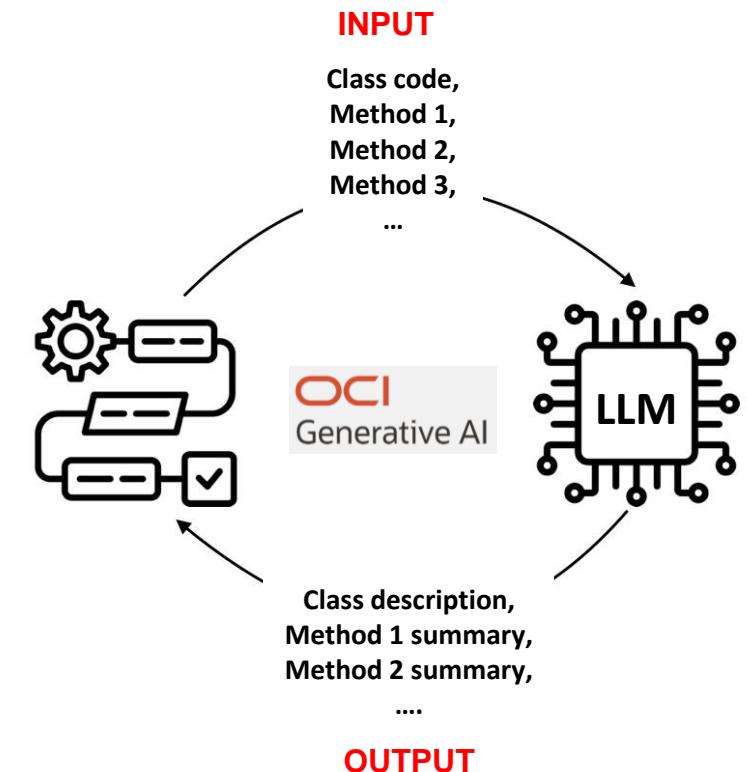
```

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()
39    Set ..AdmissionStatus = "Pending"
40
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(VALUESLIST="Pending,Active,Discharged,Canceled") [ 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 = $ZDATE($HORLOG,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.

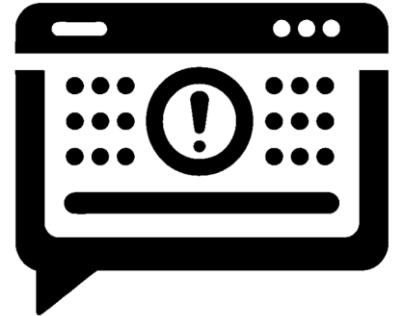
Hospital.PatientAdmission: FinalizeBilling

Method signature

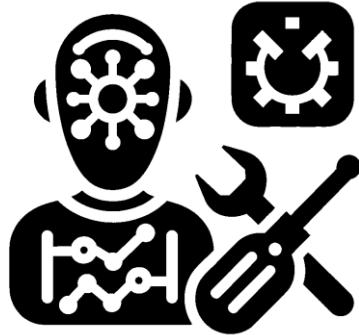
- Parameters: None
- Return type: %Status

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



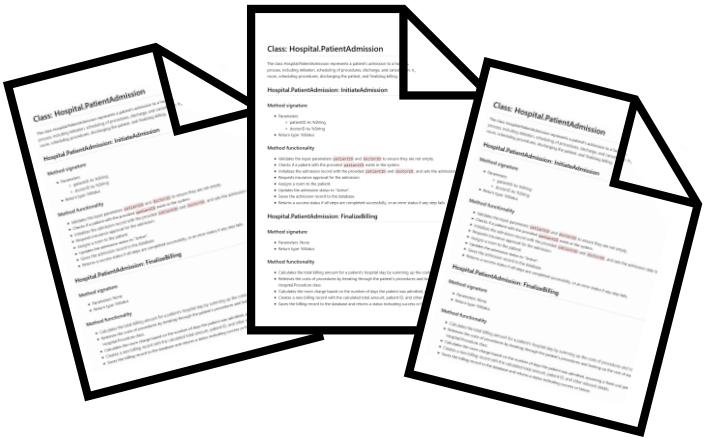
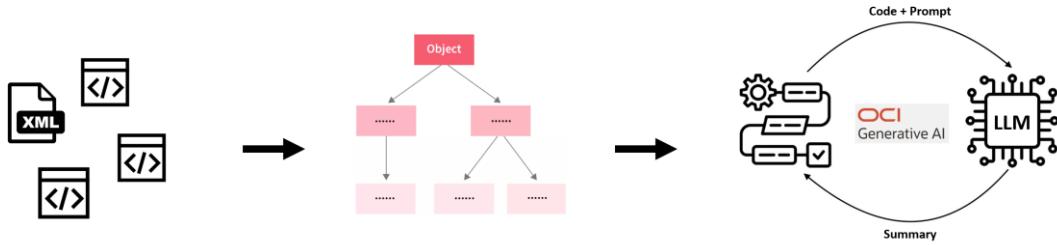
Model and prompt tuning

Context information and LLM costs

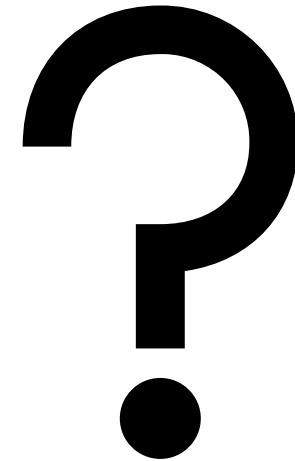


Code summaries validation

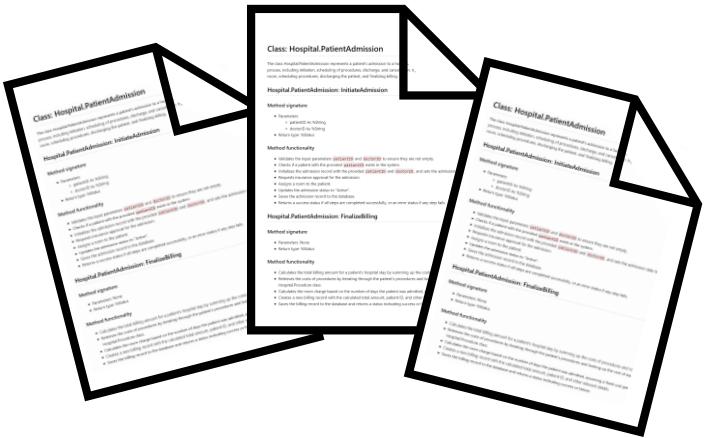
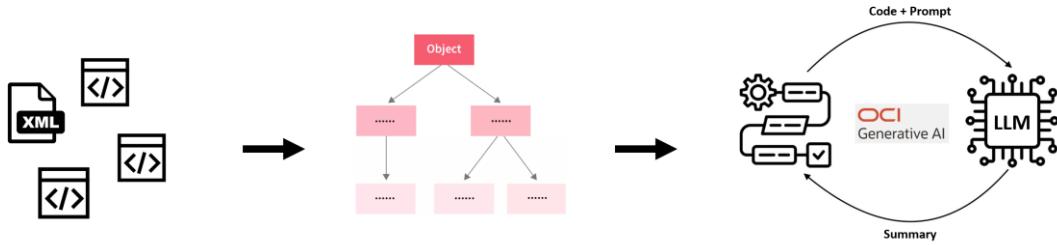
Manual validate a small sample



Extracted codebase
knowledge



What's next?



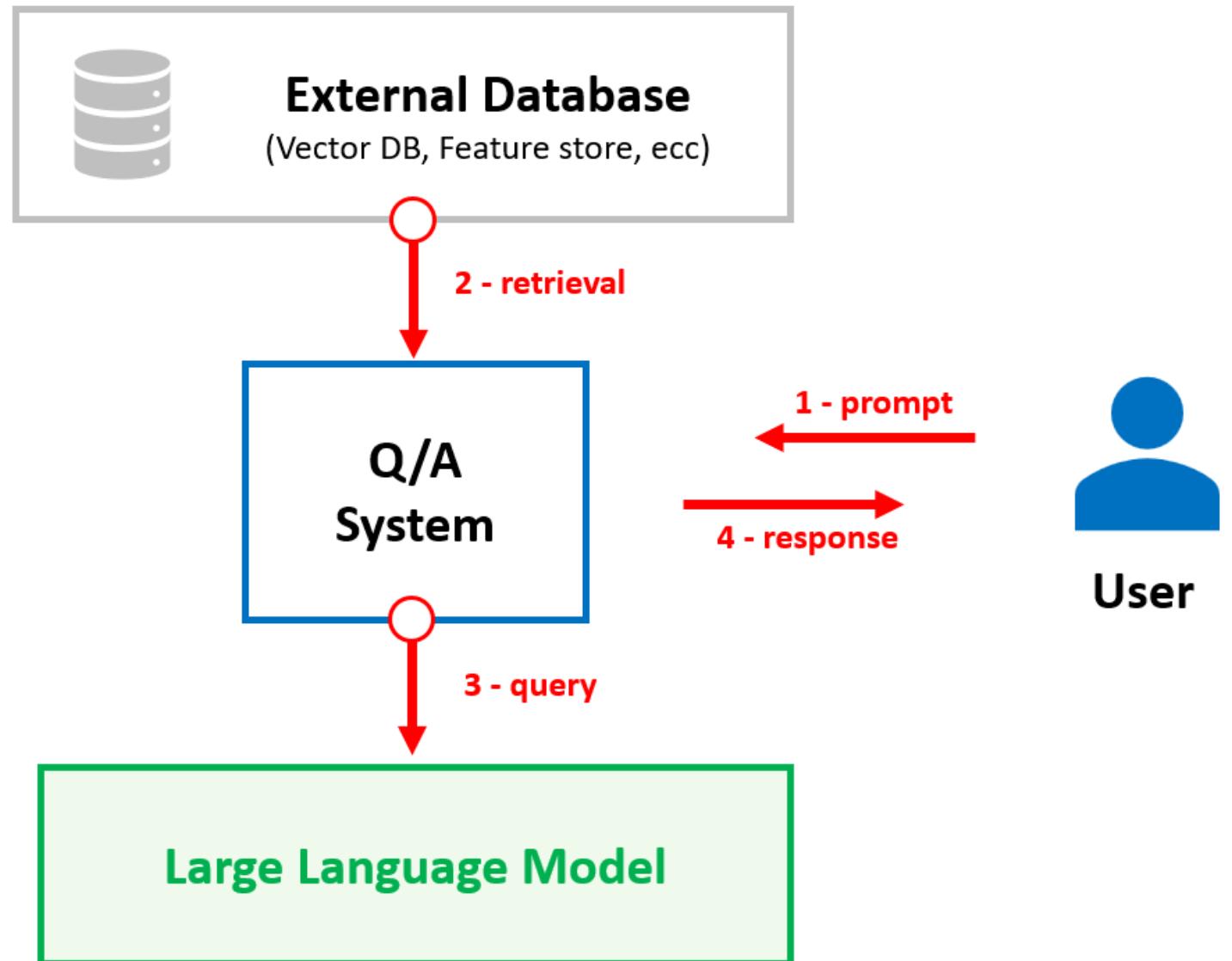
Extracted codebase
knowledge

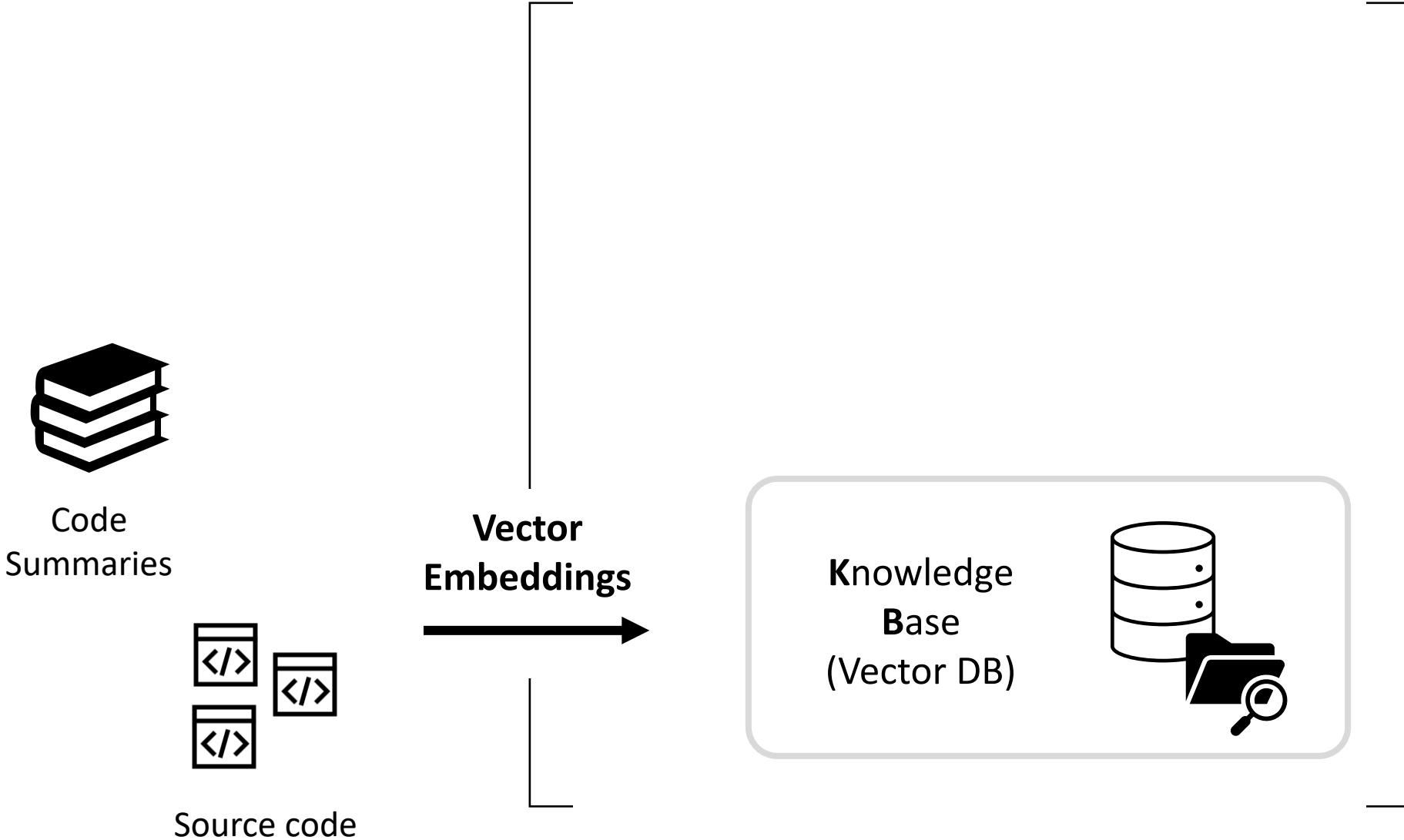


Q/A assistant
with codebase knowledge

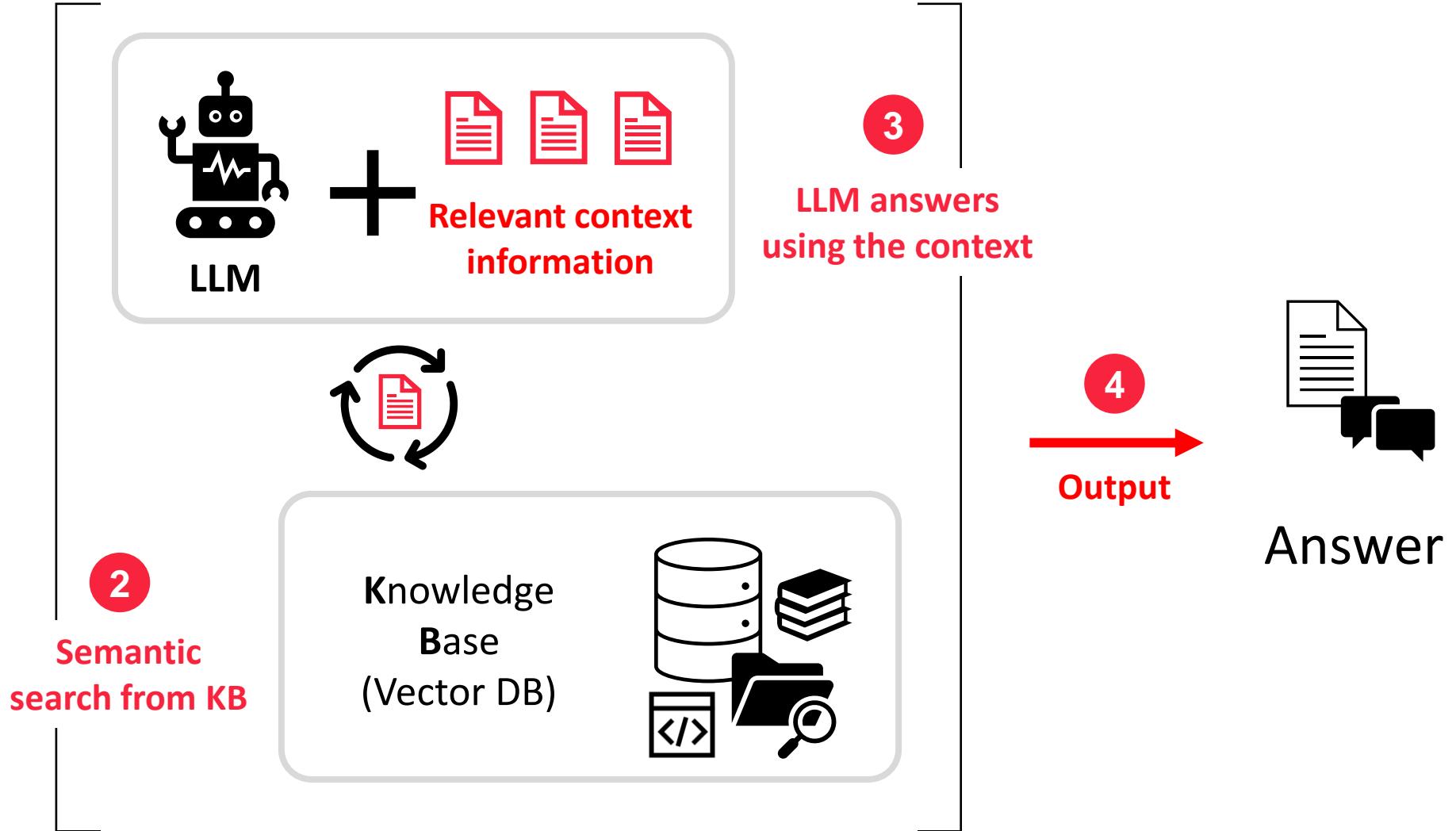
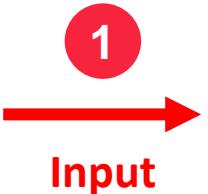
What's next?

Retrieval Augmented Generation

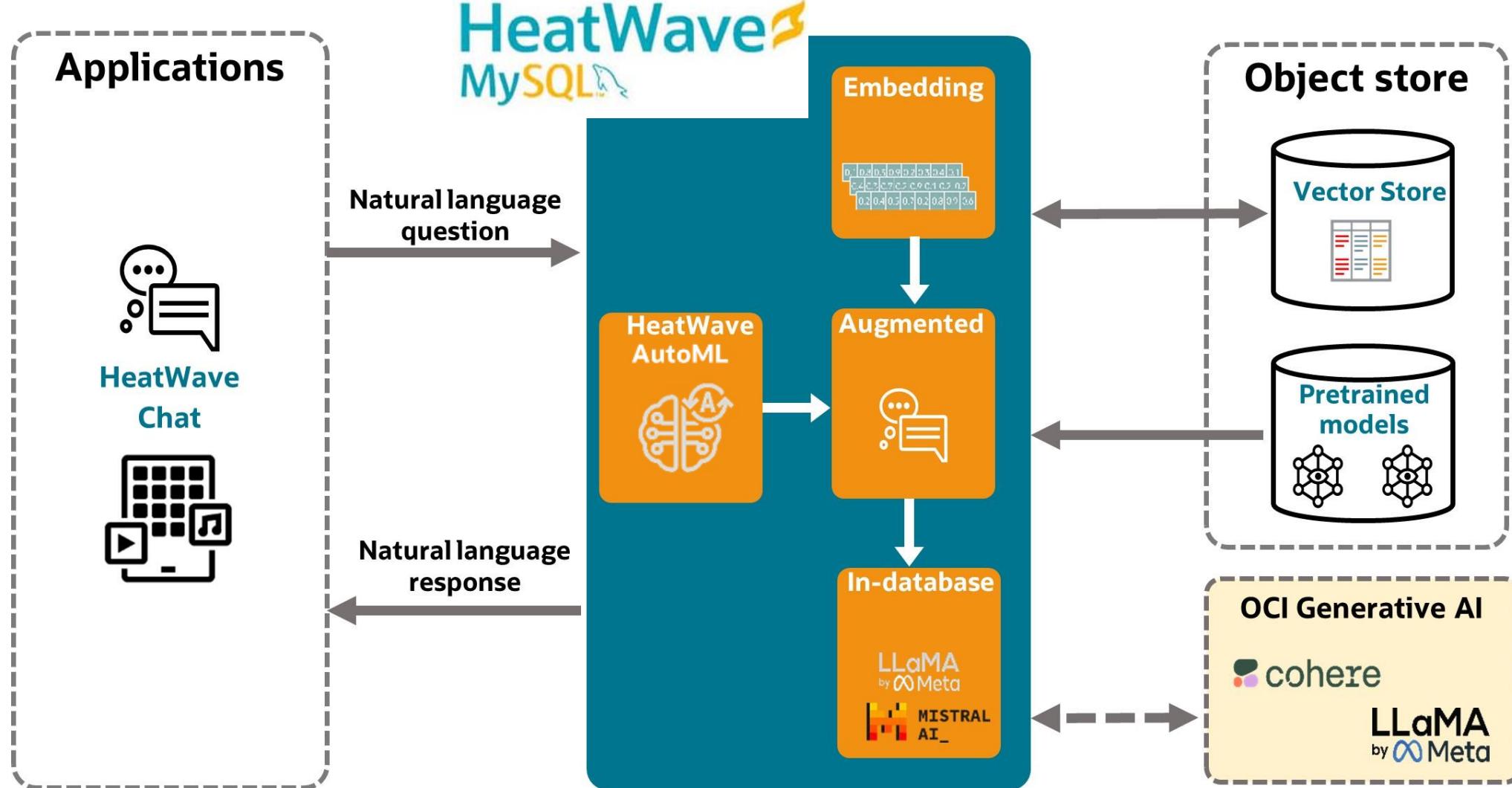




Question



RAG Q/A



Source: blogs.oracle.com

RAG Q/A using MySQL HeatWave

1

Load documents in
OCI Object Storage



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

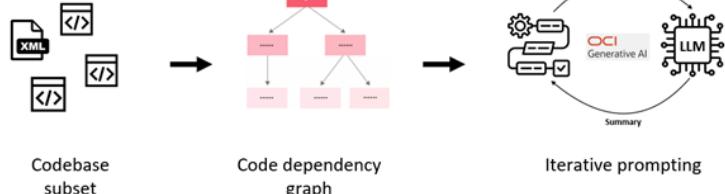
November
2022



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



Q/A assistant
with codebase knowledge



Extracted codebase
knowledge

What's next?



Giovanni Rosa, PhD
AI Engineer @Technology Reply
 giovannirosa.com

Thank You!

