



# Chat with me and get the PromQL you want

Kunyi Chen

Observability Team, Cloud Native

27<sup>th</sup> Sep 2023

# About Me

## *Application Real-time Monitoring Service (ARMS)*

**Focus on:**

**Abnormal detection,  
Root Cause Analysis**

**AIOps services in ARMS:  
Insights, Smart alert,  
Query assistant**



**Kunyi Chen**  
Alibaba Cloud  
Algorithm Engineer  
hangzhou

Kunyi Chen is currently working as an Algorithm Engineer with the Cloud Native Observability Group at Alibaba Cloud. She has presented her research at the CCF A conference INFOCOM.

She completed her Ph.D. from the Department of Computing at Harbin Institute of Technology in July 2022. During her studies, she was also a visiting Ph.D. student at Imperial College London, U.K from November 2020 to December 2021.

[Chenkunyi.cky@Alibaba-inc.com](mailto:Chenkunyi.cky@Alibaba-inc.com)

# Outlines

---

- Why do we need a natural language to PromQL chatbot
- The road we have taken
- Where are we now
- The road ahead

# Why do we need a natural language to PromQL chatbot ?

## *What is PromQL ?*

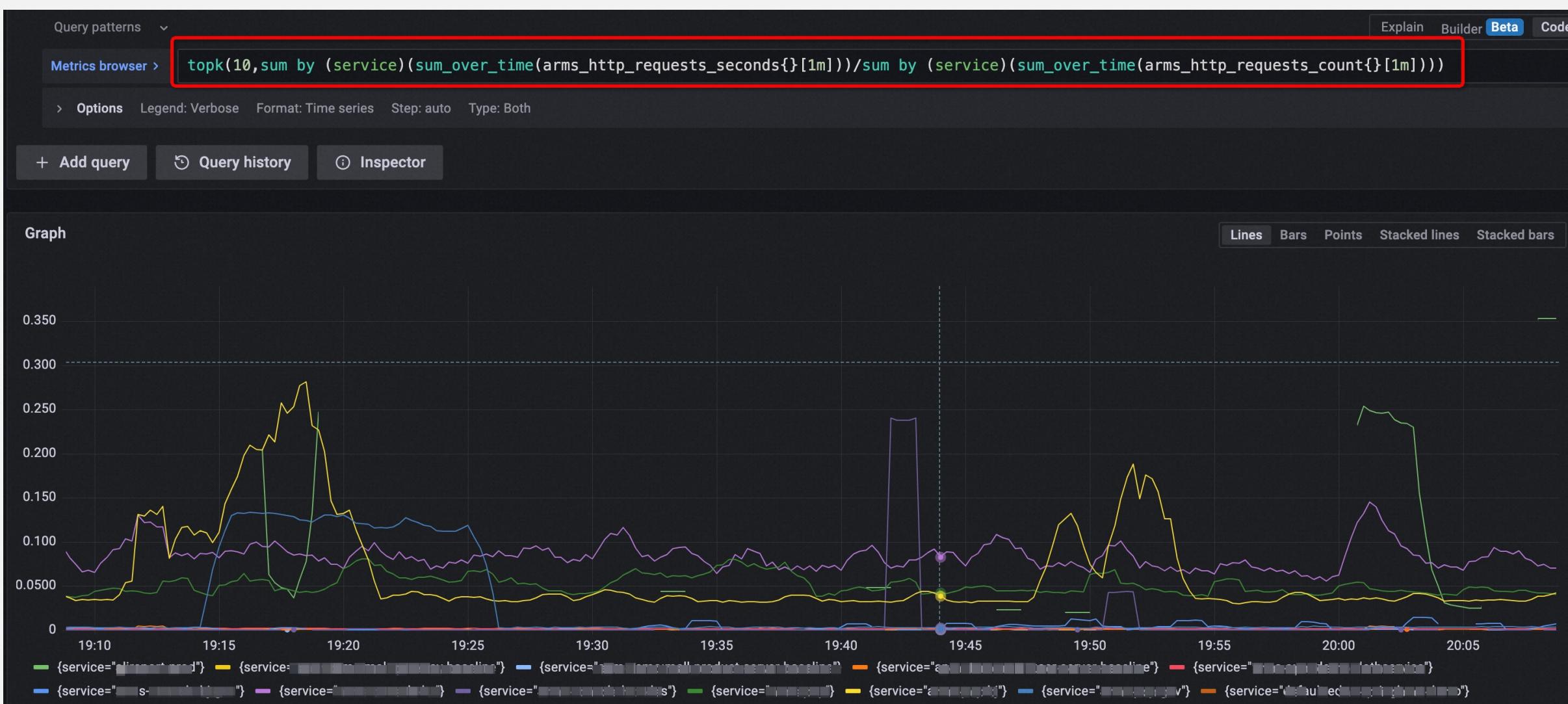
- **PromQL:** Query language of Prometheus, totally different from conventional SQL
- **Prometheus:** “De facto standard” for K8s application metrics

## *Why do we need a chatbot?*

**Complex syntax**

**Confusing metric name**

**Commonly used query**



# Why do we need a self-built chatbot ?



Write a PromQL and show me the top 10 applications with the highest average response time in the past 5 minutes



ChatGPT3.5

To find the top 10 applications with the highest average response time in the past 5 minutes using PromQL, you can use the following query:

```
topk(10, avg_over_time(application_response_time_seconds[5m]))
```

In this query, `application_response_time_seconds` is the metric representing the response time of each application. `avg_over_time` calculates the average response time over a 5-minute period. The `topk` function retrieves the top 10 applications based on the average response time.

Please note that you need to replace `application_response_time_seconds` with the actual metric name used in your Prometheus setup to measure the response time of applications.



Query patterns ▾

Metrics browser > `topk(10, avg_over_time(application_response_time_seconds[5m]))`

> Options Legend: Verbose Format: Time series Step: auto Type: Both

+ Add query ⏪ Query history ⓘ Inspector

No data

## General LLMs:

- Have some knowledge in PromQL syntax ✓

- Lack of knowledge about the ARMS system:

**Metric name, Users intention**

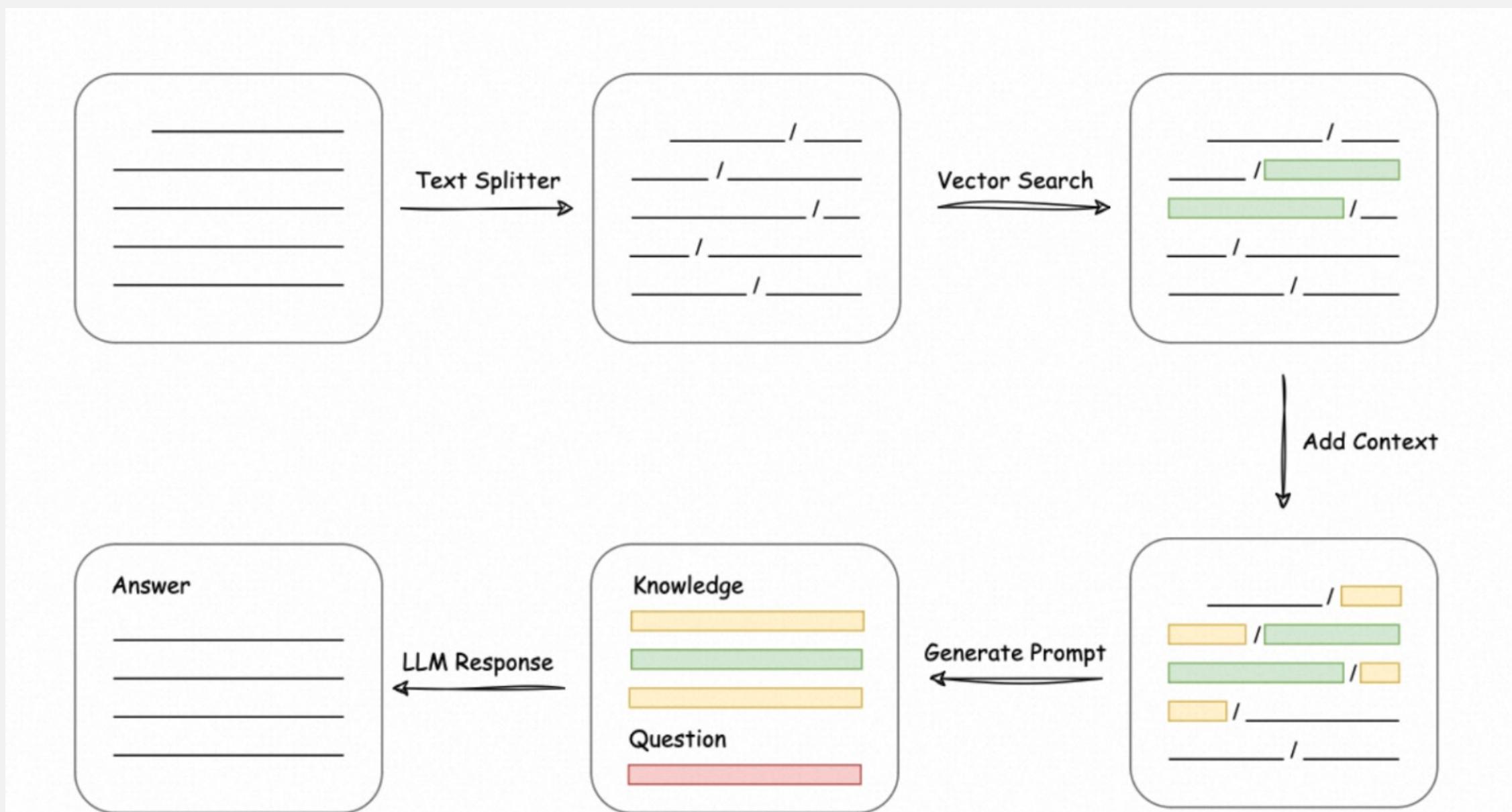
LLM needs some knowledge about our system...



# The road we have taken

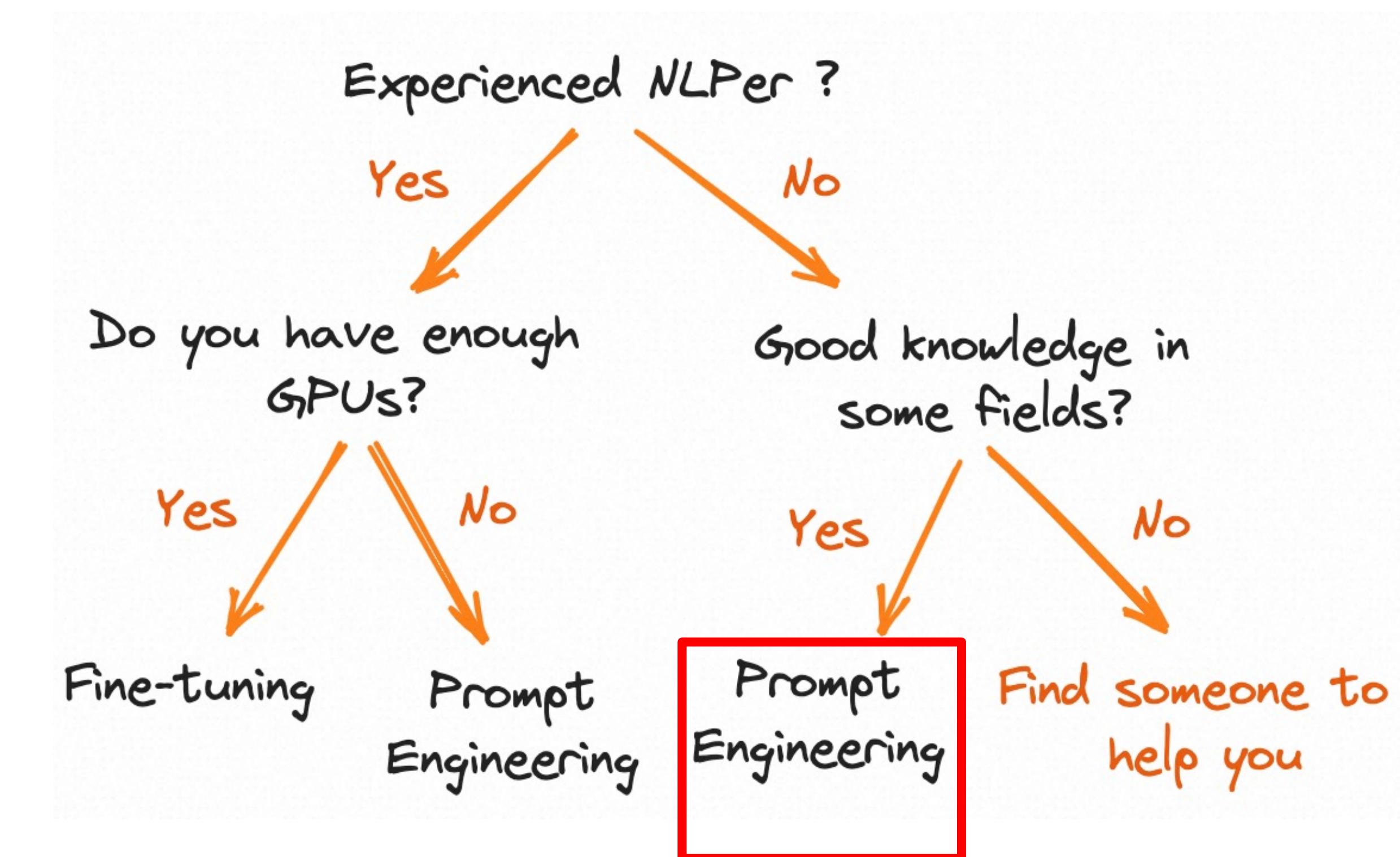
# Fine-tuning or Prompt Engineering ?

- **Fine-tuning:** Build your own LLM with enough corpus
- **Prompt Engineering:** Use an existing LLM without modifying any parameters, but provide prompts
- **Prompt:** the additional knowledge that accompanies your question **when it is being inputted into an LLM**



Langchain + ChatGLM Made by imClumsyPanda

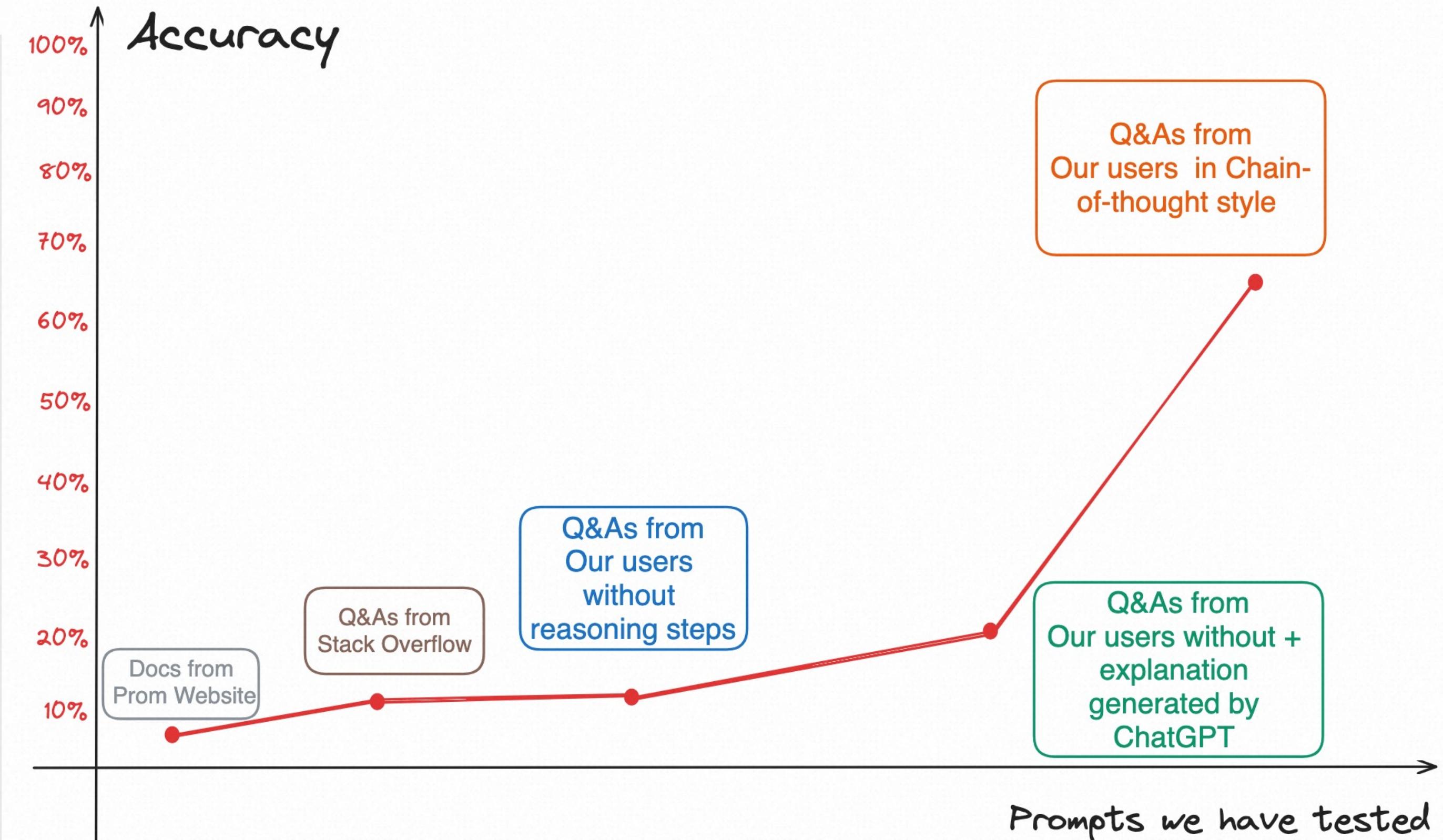
**A decision tree for the beginners in LLM-based service**



# What kind of prompt works ?

Alibaba Cloud

Prompts	Analysis
Documents about PromQL from Prometheus official website	<ol style="list-style-type: none"><li>1. Metric names are different from those used in our system</li><li>2. The documents are too long and LLM can not get the point</li></ol>
Q&A about PromQL from Stack Overflow (a community platform for developers)	<ol style="list-style-type: none"><li>1. Metric names are different from those used in our system</li><li>2. Many of the answers are incorrect</li><li>3. Some of the functions are outdated and invalid</li></ol>
Q&As collected from our users, which lack reasoning steps	<ol style="list-style-type: none"><li>1. PromQL has a complex syntax</li><li>2. Answers that lack reasoning steps are insufficient for the LLM</li></ol>
Q&As from our users + Explanation of PromQL that generated by ChatGPT	<ol style="list-style-type: none"><li>1. These explanations are not always accurate.</li><li>2. Such prompts are insufficient for the LLM to generate accurate promQL</li></ol>



arXiv  
https://arxiv.org > cs ::

Chain-of-Thought Prompting Elicits Reasoning in Large ...

作者: J Wei · 2022 · 被引用次数: 1169 — Experiments on three large language models: that chain of thought prompting improves performance on a range of arithmetic,...

您已浏览过该网页 4 次。上次访问日期: 2023/8/9



# What is Chain-of-Thought Prompt

Chain-of-  
Thought  
Prompt is  
knowledge  
with  
reasoning  
steps

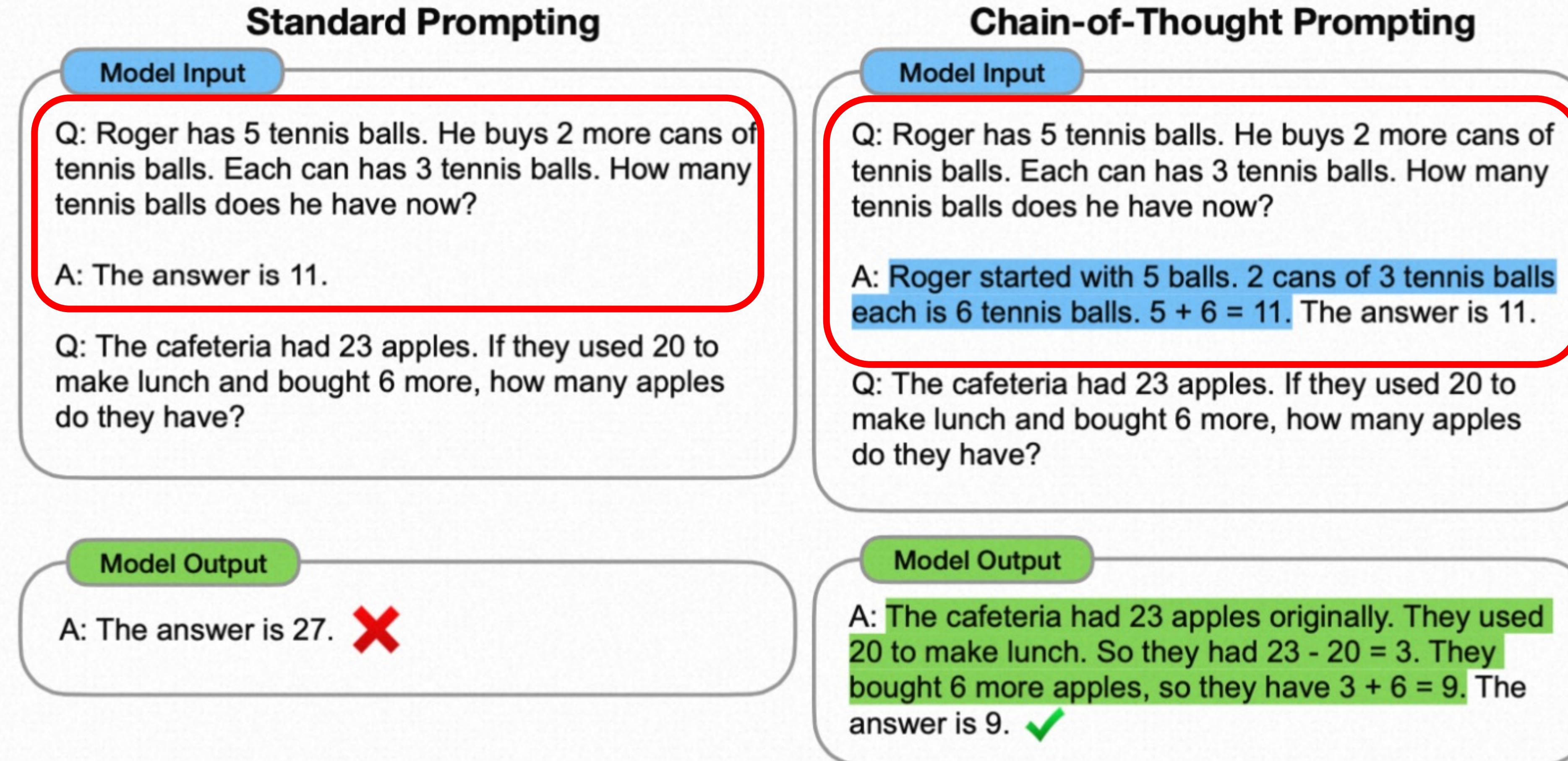


Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

# Chain-of-Thought Prompt in natural language to PromQL

Q: Write a PromQL and show me the top 10 applications with the highest average response time in the past 5 minutes.

A: Firstly, we need to calculate the total latency of each application in the past 5 minutes with the following PromQL:

`"sum by (service)(sum_over_time(arms_http_requests_seconds{}[5m]))"`

1

Secondly, calculate the total number of calls for each application with the following PromQL:

`"sum by (service)(sum_over_time(arms_http_requests_count{}[5m]))"`

2

Thirdly, get the average response time of each service. The PromQL is as follows:

`" sum by (service)(sum_over_time(arms_http_requests_seconds{}[5m]))/sum by (service)(sum_over_time(arms_http_requests_count{}[5m])) "`

3

=

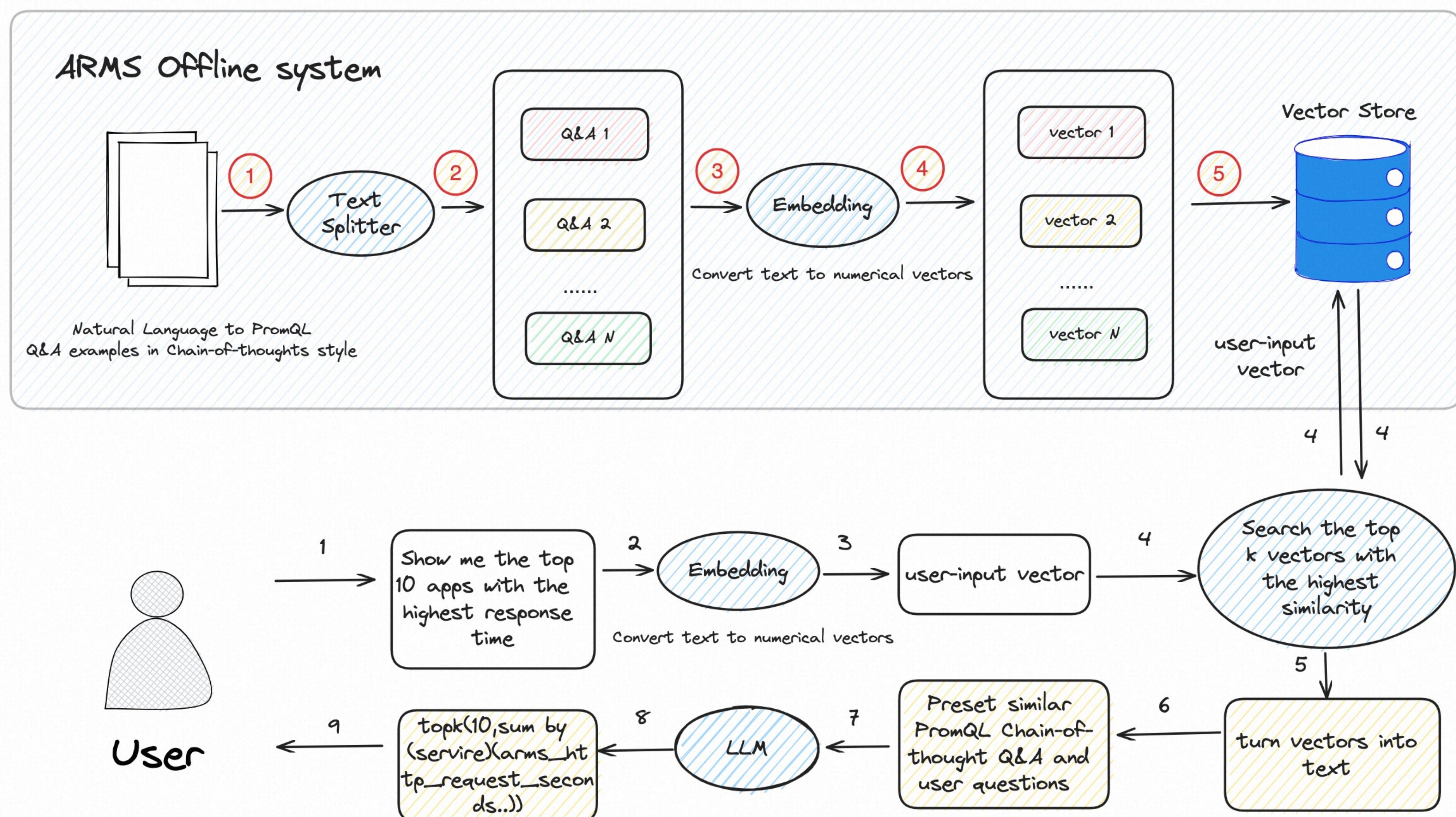
1 / 2

Finally, use the topk function to find the top 10 applications with the longest response time.

The PromQL is as follows:

`"topk(10,sum by (service)(sum_over_time(arms_http_requests_seconds{}[5m]))/sum by (service)(sum_over_time(arms_http_requests_count{}[5m])))"`

# Architecture of natural language to PromQL Chatbot



# Demo

CC · 账号管理 | 阿里云控制台首页 | Explore - arms\_metrics\_cn-ha | 应用实时监控服务ARMS | +

arms.console.aliyun.com/?spm=5176.12818093.top-nav.14.5adc16d0E6mgPv#/home

阿里云 工作台 搜索... 费用 ICP 备案 企业 支持 工单 APP 通知 购买 简体 E-chenkunyi... RAM 用户

### 接入中心

支持40+应用类型，满足各类监控场景 [查看全部→](#)

#### 可观测可视化 Grafana 版

总数 34 [查看全部](#)

工作区ID/名称	地域	运行状态	访问地址url
Grafana 共享版	运行中		
g-arg97mhs16k48zguljln080	华东1 (杭州)	停机	http://g-arg97mhs16k48zguljln080.grafana.aliyuncs.com:30
cjtest			
grafana-rnggphacnqkywy73u6	华东1 (杭州)	停机	https://grafana-rnggphacnqkywy73u6.grafana.aliyuncs.com:443
kefantest-0127			
grafana-rnggywlumn1c3ypdi	华东1 (杭州)	停机	http://grafana-rnggywlumn1c3ypdi.grafana.aliyuncs.com:80

#### 报警

总数 0 [查看全部](#)

您的应用非常健康，目前没有任何报警。

#### 应用监控

总数 399 报警 0 [查看全部](#)

应用名称	地域	每秒请求数	错误率	平均响应时间	智能洞察
arms-console-bj-gov	华东1 (杭州)	41.80	0.02%	1.8ms	1
arms-console-hz-edas	华东1 (杭州)	1.03	1.29%	0.8ms	0
aia	华东2 (上海)	0.00	0.00%	0ms	0
backend	华东2 (上海)	0.00	0.00%	0ms	0
TEST	香港	0.00	0.00%	0ms	0
TEST1	香港	0.00	0.00%	0ms	0

### 资源状况

产品 当前版本 操作

应用监控 专家版 [购买资源包](#)

资源包数量: 0 剩余资源: 0 Agent×小时

前端监控 专家版 [购买资源包](#)

资源包数量: 0 剩余资源: 0 数据上报

Prometheus监控 专家版 [创建实例](#)

实例数量: 144

云拨测 专家版 [购买预付费](#)

可用月数: 0

智能告警 专家版

可观测可视化 Grafana 版 高级版 [创建工作区](#)

应用安全 专家版 [购买资源包](#)

### 快捷方式

开始监控 [价格计算器 \(Beta\)](#)

ARMS帮助文档 [ARMS学习路径](#)

授权协助 [价格详情说明](#)

### 产品快报

欢迎进入「ARMS」，点击我立即了解 [X](#)

Grafana服务零元首月免费用！让可视化更简...

ARMS OpenTelemetry Collector正式发布

Prometheus监控云服务，基础指标免费上报！ [NEW](#)



今天20:56:11



# Where are we now ?



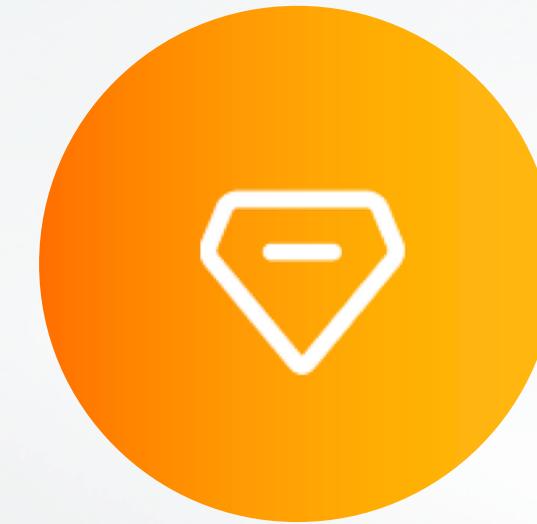
Covering 14 scenarios

Response time、error rate、qps...

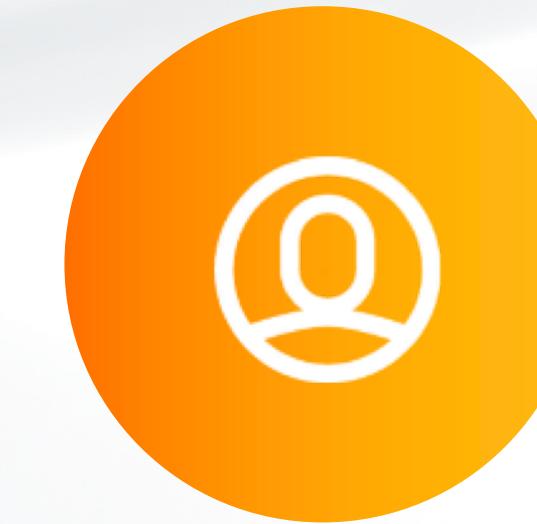


Providing useful info

Correct metric name, proper syntax



76.9% Accurate Rate



Rapid coverage of new scenarios

The only thing we need to do is to add about 5 examples



小智 今天20:56:17

Sure, here's the PromQL query to show the top 10 applications with the highest error rate:

```
topk(10, sum_by(service) (sum_over_time(arms_http_requests_error_count{service} [1h])) / sum_by(service) (sum_over_time(arms_http_requests_count{service} [1h])))
```

This query uses the `topk` function to return the top 10 applications with the highest error rate. The `sum_by` function is used to sum the error counts and error rates over time for each service. The `sum_over_time` function is used to aggregate the data over time, with the argument `[1h]` indicating that we want to look at data from the past hour. Finally, the division operator `/` is used to divide the error rates by the error counts to get the error rate for each application. I hope this helps! Let me know if you have any further questions.

如果您需要人工服务，欢迎直接联系ARMS服务号 [【ARMS/链路追踪服务-七七】](#)（钉号：arms160804）与应用实时监控服务工程师即时沟通。



小智 今天15:04:53

PromQL小工具

ARMS 为您提供以下- 场景 PromQL 语句：

- 场景1: 响应时间相关
- 场景2: 调用次数相关
- 场景3: 调用次数降量相关
- 场景4: 调用次数增量相关
- 场景5: 错误率相关
- 场景6: 错误数相关
- 场景7: HTTP状态码相关
- 场景8: CPU使用率相关
- 场景9: JVM\_GC\_次数相关
- 场景10: JVM\_GC\_耗时相关
- 场景11: 慢SQL相关
- 场景12: ARMS中的控制台\_累计错误数
- 场景13: ARMS中的控制台\_累计请求数
- 场景14: ARMS中的控制台\_累计异常数

如果您需要人工服务，欢迎直接联系ARMS服务号 [【ARMS/链路追踪服务-七七】](#)（钉号：arms160804）与应用实时监控服务工程师即时沟通。

# ChatGPT

# ChatGPT3.5 without prompts

 Write a PromQL and show me the top 10 applications with the highest average response time in the past 5 minutes

# ARMS Query Assistant

# Qwen-turbo with 3-5 prompts

今天20:55:48 

write a promql, show me the top 10 applications with highest response time

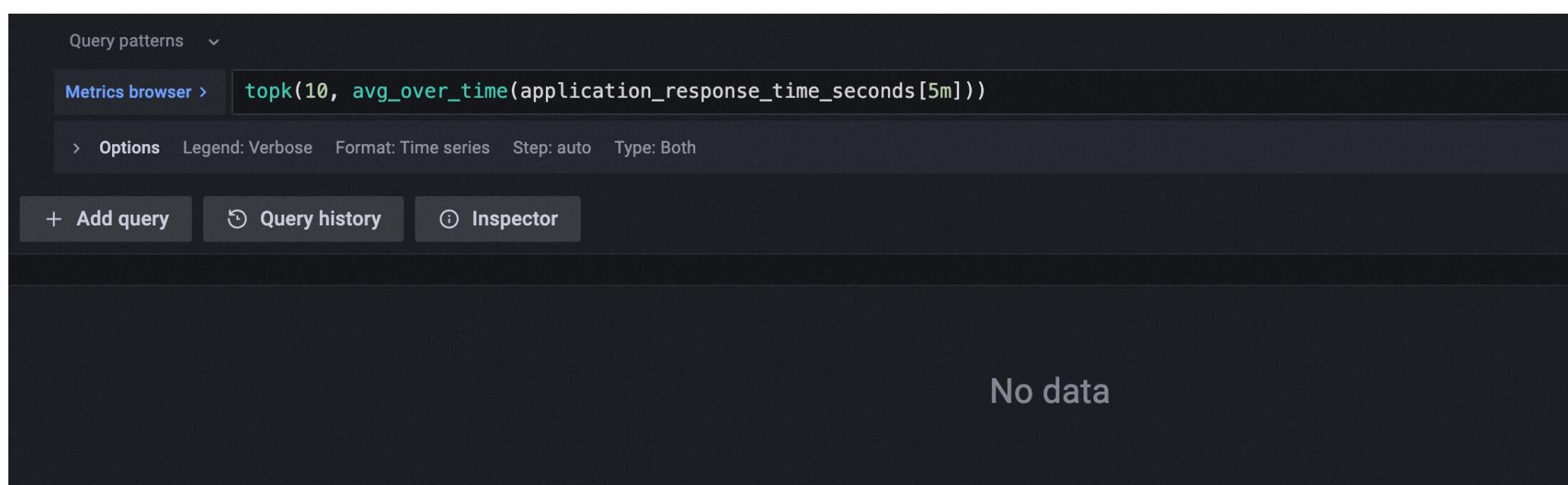
小智 今天20:55:53 

To find the top 10 applications with the highest response time, you can use the following PromQL:

```
topk(10, sum_by(service)(sum_over_time(arms_http_requests_seconds{}[10m])) / sum_by(service)(sum_over_time(arms_http_requests_count{}[10m])))
```

This PromQL calculates the sum of response time over the last 10 minutes for each application, and then takes the top 10 applications based on that sum. The result will show the application name and the corresponding response time sum.

如果您需要人工服务，欢迎直接联系ARMS服务号 【ARMS/链路追踪服务-七七】（钉号：arms160804）与应用实时监控服务工程师即时沟通。



## Syntax

1

# Metric name

X

# Intention recognition

X

## Syntax

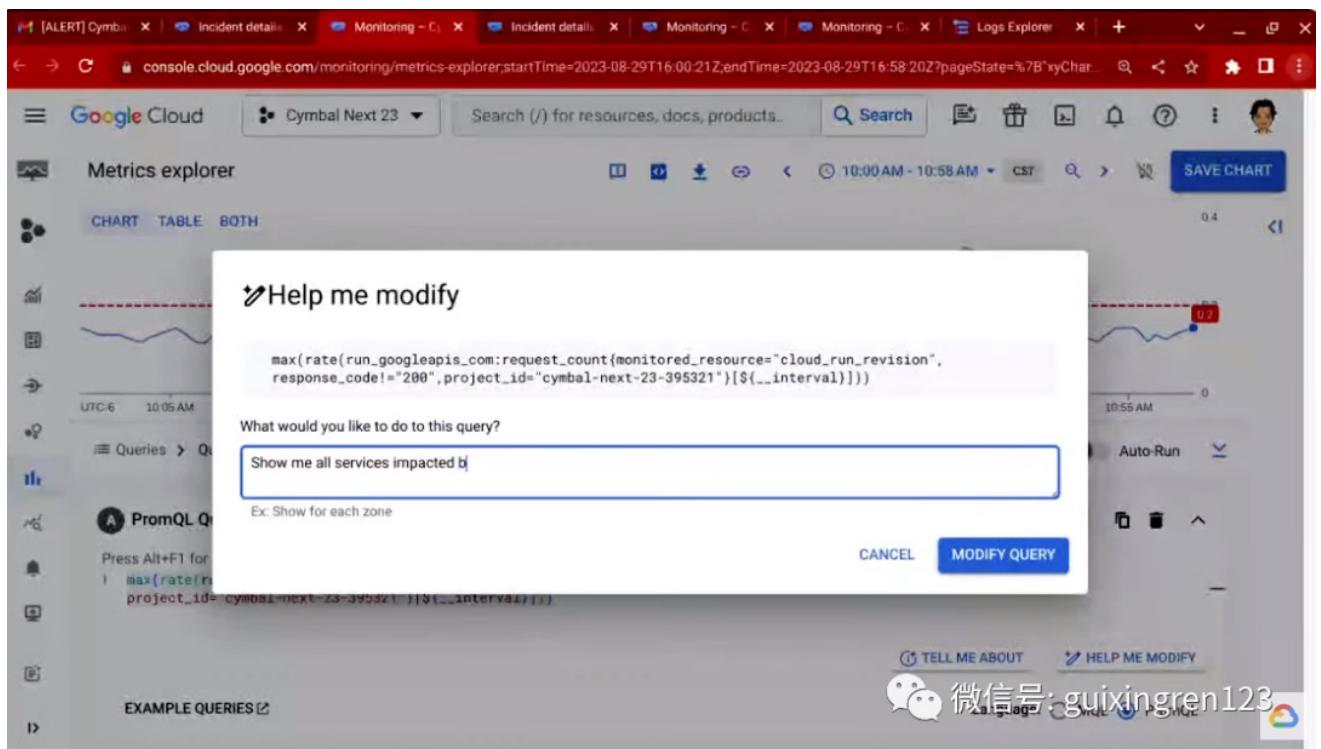
1

# Metric name

1

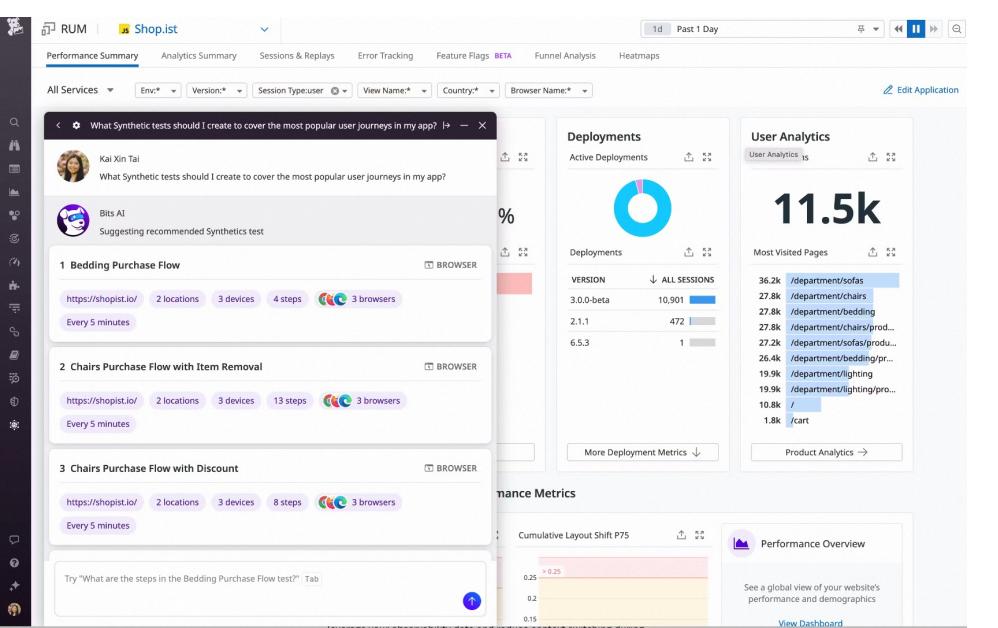
# Intention recognition

1



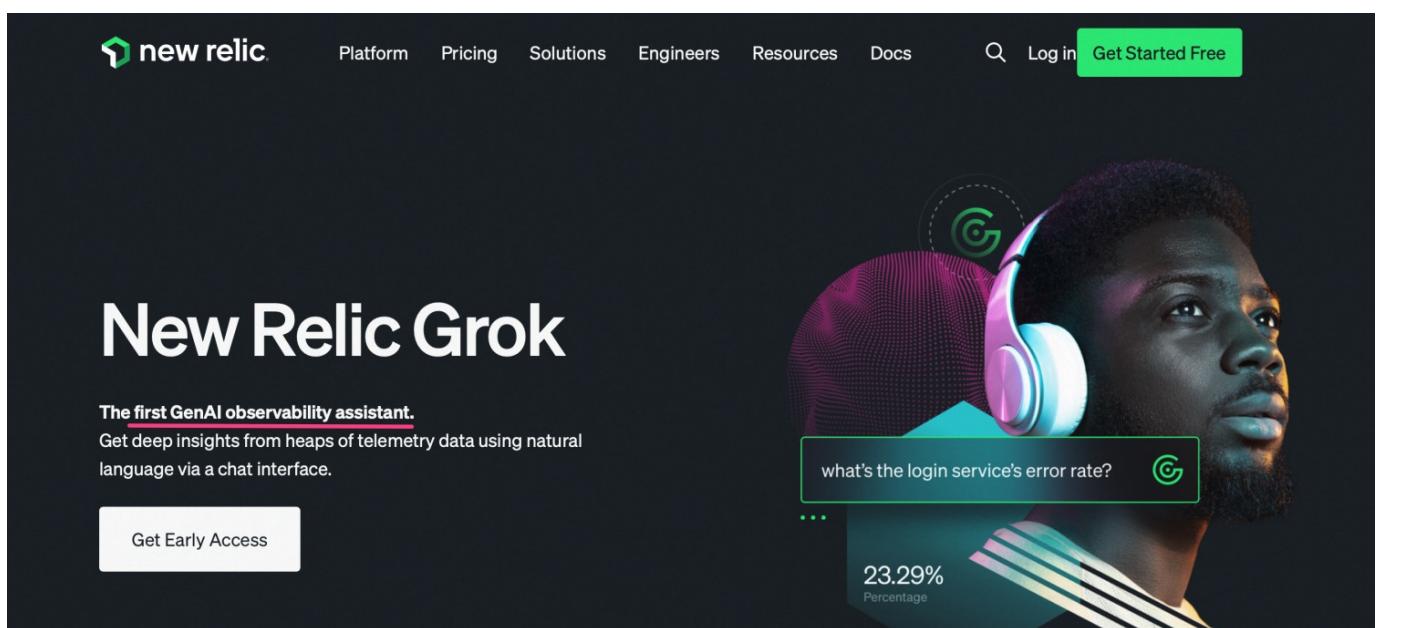
A screenshot of the Google Cloud Metrics explorer. It shows a PromQL query editor with a query: `max(rate(run.googleapis.com:request_count{monitored_resource="cloud_run_revision", response_code!="200", project_id="cymbal-next-23-395321"}[$__interval]))`. Below the query is a chart showing request count over time. A modal window titled "Help me modify" is open, asking "What would you like to do to this query?" with options like "Show me all services impacted by this". There are also "TELL ME ABOUT" and "HELP ME MODIFY" buttons.

## Google Duet AI



A screenshot of the Datadog Bits AI interface. It shows a dashboard with several monitoring panels. One panel displays deployment metrics for a "Bedding Purchase Flow" with 11.5k users. Another panel shows session replays for a "Chairs Purchase Flow with Item Removal". A third panel shows a "3 Chairs Purchase Flow with Discount". The interface includes filters for environment, version, session type, user, view name, country, and browser name.

## Datadog Bits AI



A screenshot of the New Relic Grok interface. It features a dark-themed chat window where a user asks "what's the login service's error rate?" and Grok responds with "23.29% Percentage". Below the chat is a CPU Usage (%) chart showing a sharp spike from 11:30 am to 12:00 pm. The interface includes a "Get Early Access" button and a "New Relic Grok" header.

## New Relic Grok

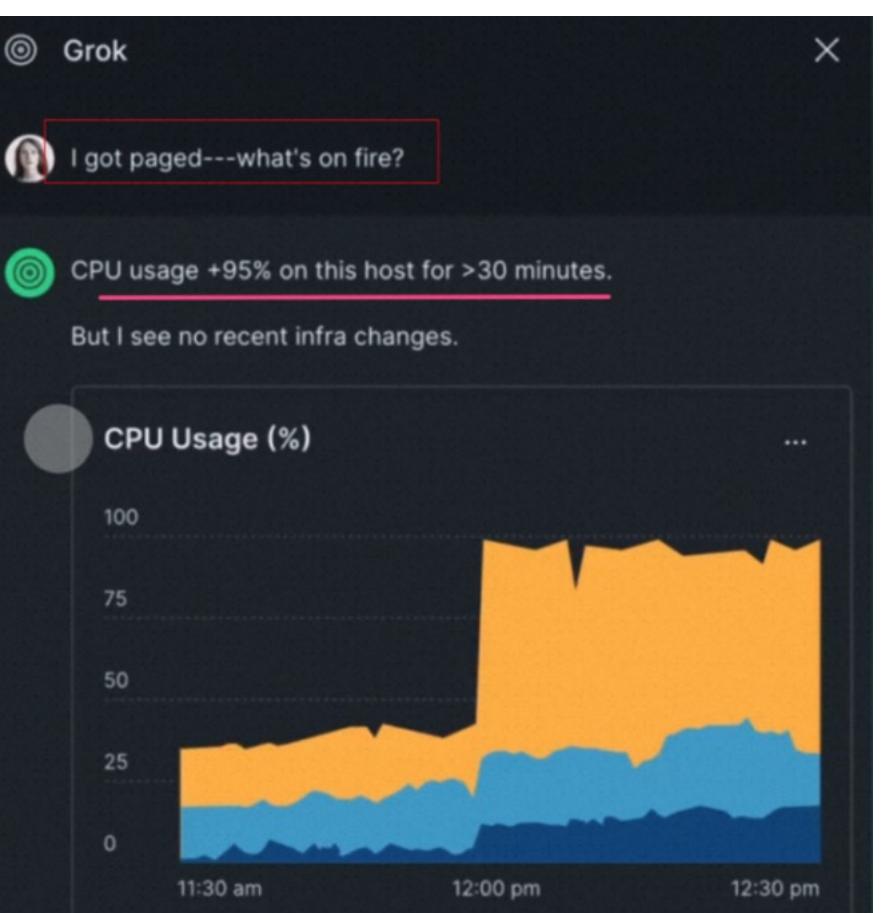
# The Road Ahead

## Chatbots for monitoring

Chat with me and get the alerts and fixed suggestions your want

## Natural Language to chart

Chat with me and get the chart you want



 **Grok** 1 minute ago  
The error is caused by the 'get' method calling an object that's undefined.  
  
To fix this, add a return statement to the reduce function:  
  
01. function countUsersByState() {  
02. return userData.reduce((map, user) => {  
03. const count = map.get(user.address.state);  
04. if (count) {  
05. map.set(user.address.state, count + 1);  
06. } else {  
07. map.set(user.address.state, 1);  
08. }  
09. }, new Map());  
10. }  
11. console.log(countUsersByState());

## Chat Ops

Chat with me and I will take care of your systems and applications



## Takeaways

*Chain-of-Thought Prompt  
works*

*Q&A examples with  
step-by-step PromQL  
generation are effective*

Thanks for your time  
and enjoy your  
holiday : )



陈昆仪  
浙江 杭州



扫一扫上面的二维码图案，加我为朋友。