AiDD

# 2024 AI+研发数字峰会

AI+ Development Digital summit

AI驱动研发变革 促进企业**降本增效**

北京站 08/16-17

# 打造自适应AI运维智慧体：大语言模型在软件日志运维的实践

刘逸伦 华为2012实验室

# 演讲嘉宾



## 刘逸伦

华为2012文本机器翻译实验室工程师

本科毕业于南开大学，硕士毕业于美国佐治亚理工学院。研究方向包括AI智能运维，大模型质量评估以及大模型提示策略，在相关领域以第一作者、通讯作者身份在ICDE、ICSE、IWQoS等顶级国际会议/期刊发表10余篇论文。

# 目录
# CONTENTS

# PART 01

# 软件日志运维观点：

## 智能运维演进趋势是从任务数据驱动到自适应运维智慧体

# 观点1：软件日志运维是从机器语言向自然语言的转化

(1)**日志是机器语言**：大规模网络、软件系统在运行过程中每天会产生PB级别的日志，这些日志是一些类自然语言的文本，实时描述了设备的运行状态、异常情况。

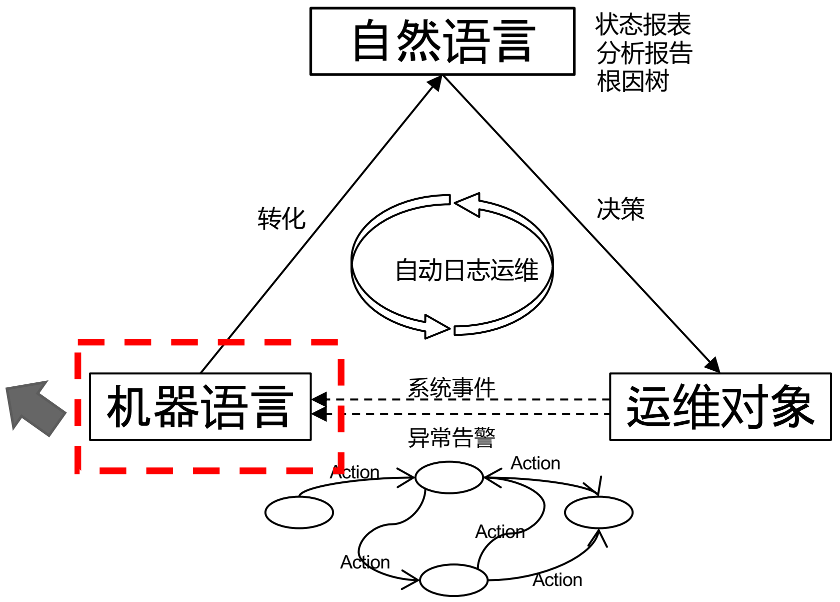
(2)**传统网络运维是机器语言的人工翻译过程**：为了维护网络的稳定，运维人员会持续监控设备的运行状态，希望准确、及时地检测异常和突发事件。网络日志是设备运行维护最重要的数据源，运维人员通常会通过解读日志中的自然语言、语义信息来发现问题、分析根因。

(3)**自动日志分析是机器语言的自动翻译过程：**日志文本种类繁多、数量庞大，且多数日志为非结构化文本，无法通过人工方式监控和检测全部的日志。更重要的是，分析设备日志需要丰富的领域知识，耗时耗力；简单的规则配置也无法理解文本的语义信息。

| 数据来源 | 时间戳 | 详细信息 |
|---|---|---|
| 交换机 | Jun 12 19:03:27 | Interface te-1/1/59, changed state to down |
| 路由器 | Jun 13 20:22:03 | GigabitEthernet1/0/18 link status is DOWN |
| 分布式文件系统 | Jun 12 19:03:27 | PacketResponder 1 for block blk_1608 terminating |
| 高性能计算集群 | Jun 12 19:03:27 | Component is in the unavailable state(HWID=4432) |

类自然语言半结构化文本

表：一些网络基础设施中的日志消息，日志中的详细信息和自然语言有一定的相似性



# 观点2：智能运维演进趋势：从任务数据驱动到自适应运维智慧体

| 代际 | 输入 | 方法 | 目标 | 研究成果 | 类别 |
|---|---|---|---|---|---|
| 第一代 | 离散特征和KPI | 特征识别及统计算法 | 拟合异常结果 | Ft-tree<br>LogParse | 任务数据驱动 |
| 第二代 | 日志文本生成token | 深度学习 | 拟合异常结果 | LogAnomaly<br>LogStamp | |
| 第三代 | 段落日志和跨域日志 | 预训练语言模型 | 日志语言理解 | BigLog<br>Da-Parser | |
| 第四代 | 原始日志和自然语言文本 | 大语言模型 | 可解释性运维 | LogPrompt | 指令驱动 |
| 第五代 | 自适应运维智慧体：目标自适应、领域自适应、强交互性、可执行性。。。 | | | | |

表：LogAIBox研究项目代际演进思路


[1]LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs (IJCAI 2019)
[2]LogParse: Making Log Parsing Adaptive through Word Classification. (ICCCN 2020)
[3]LogStamp: Automatic Online Log Parsing Based on Sequence Labelling. (WAIN Performance 2021)
[4]BigLog:Unsupervised Large-scale Pre-training for a Unified Log Representation. (IWQoS 2023)
[5]DA-Parser: A Pre-trained Domain-Aware Parsing Framework for Heterogeneous Log Analysis. (COMPSAC 2023)
[6] Logprompt: Prompt engineering towards zero-shot and interpretable log analysis. (ICSE 2024 & ICPC 2024)

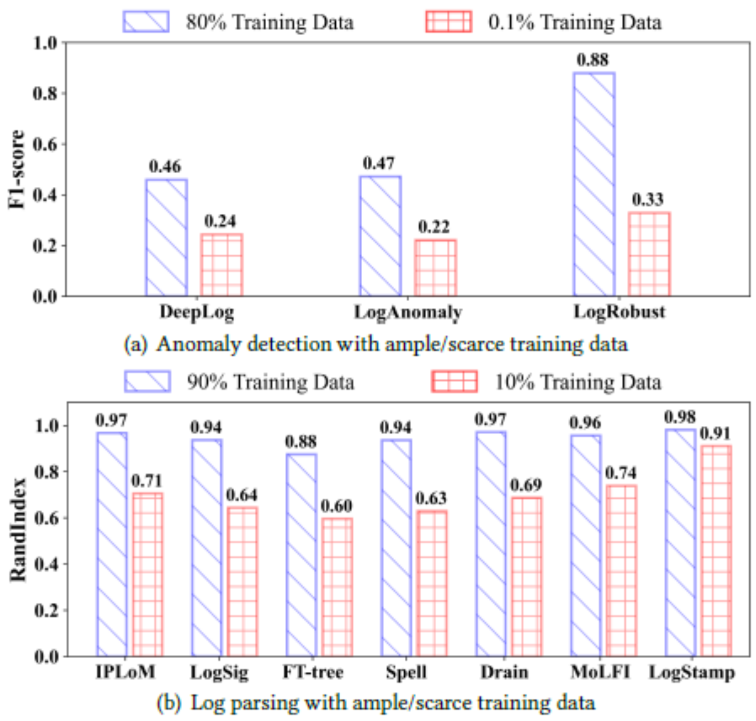
团队repo地址：https://github.com/LogAIBox

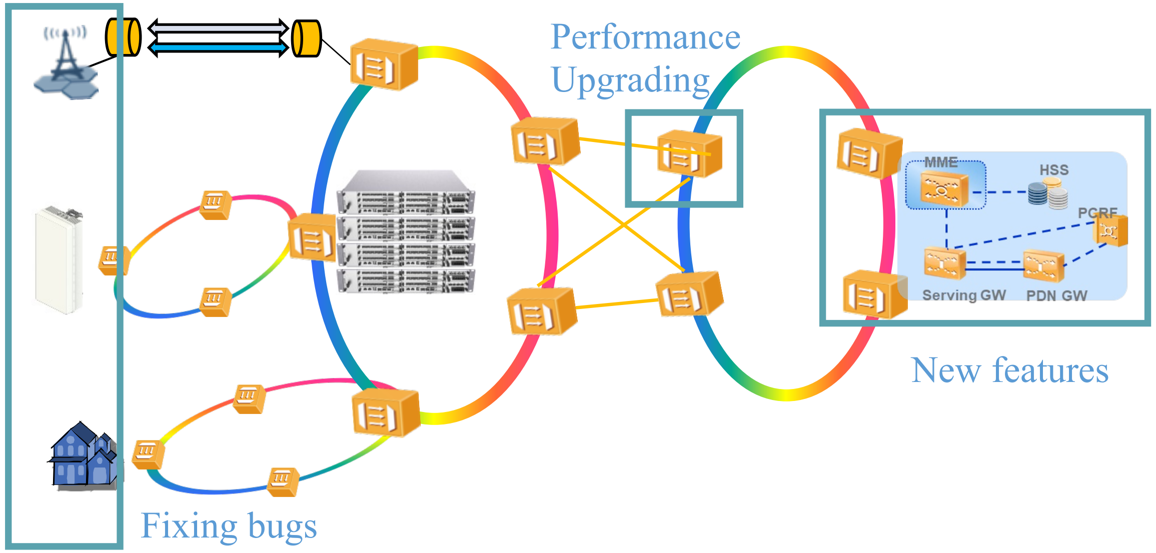
# PART 02

# 自适应智慧体在运维领域面临的Gap：

传统自动运维模型既没法“自适应”，也仅是有限“智慧”

# Gap1: 传统智能运维算法依赖于任务标注数据，仅仅是拟合数据，对于新领域无法自适应




当任务训练数据减少时，传统方法普遍出现了预测精度下降。因此，要将其应用到私有系统中，必然需要大量标注数据。

在线场景下，由于频繁的软件更新、第三方插件等，大部分产生的日志都是模型未见过的，难以获得足量的历史标注数据，需求模型有**自适应能力**。

# Gap2: 传统运维系统可解释性差、可交互性弱，智慧有限

**Raw Log**

| |
|---|
| 2005-06-03 16:47:20 63543 double-hummer alignment exceptions |
| 2005-06-03 16:56:14 162 double-hummer alignment exceptions |
| 2005-06-03 16:56:55 141 double-hummer alignment exceptions |
| 2005-06-03 18:21:59 CE sym 2, at 0x0b85eee0, mask 0x05 |
| 2005-06-04 00:24:32 ciod: failed to read message prefix on control stream (CioStream socket to 172.18.24.3:2243 |
| 2005-06-04 00:24:36 ciod: failed to read message prefix on control stream (CioStream socket to 172.16.96.1:3356 |

**Log Parsing**

| Timestamp | Template | Variable |
|---|---|---|
| 1117842440 | <*> double-hummer alignment exceptions | 63543 |
| 1117842974 | | 162 |
| 1117843015 | | 141 |
| 1117848119 | CE sym <*>, at <*>, mask <*> | 2,0x0b85eee0, 0x05 |
| 1117869872 | ciod: failed to read message prefix on control stream (CioStream socket to <*> | 172.18.24.3:2243 |
| 1117869876 | | 172.16.96.1:3356 |

**Log Anomaly Detection**

| Is Anomaly? |
|---|
| Yes. |

（1）传统日志分析算法只输出“告警/正常”，对于异常日志无反馈，需要专家阅读相关日志模板，人力整理生成分析报告，费时费力。
（2）只给出预测结果，对于报假警、漏报等情况不能很快地排除，需要结合原始日志进行分析排查。

现有方法基于任务数据可以自动映射故障现象，但依然没有完成智能运维的最后一步：根因分析和故障自恢复。**这些系统的交互设计缺乏反馈与互动，离“智慧体”距离遥远。**

**Interpretable Log Parsing**

| Timestamp | Template | Variable | Explanation |
|---|---|---|---|
| 1117842440 | ... | 63543 | Object amount: the number of instances of the "double-hummer alignment exceptions" that occurred. |
| 1117842974 | | 162 | |
| 1117843015 | | 141 | |
| 1117848119 | ... | 2 | Object name: the object name that is referenced in the log. |
| | | 0x0b85eee0 | Location indicator: the memory address of the object. |
| | | 0x05 | Other parameters: additional parameters or flags associated with the object. |
| 1117869872 | ... | 172.18.24.3 | Location indicator: the network location or address of the object. |
| 1117869876 | | 172.16.96.1 | |

对异常日志生成了解释，可以快速判断虚报、漏报。

**Interpretable Anomaly Detection**

| Is Anomaly? | Explanation |
|---|---|
| Yes. | The high number of "double-hummer alignment exceptions" and the failed attempts to read messages from a control stream suggest underlying issues with the system. |

根据本轮分析结果由大语言模型自动生成了分析报告，推荐解决方案。

**可交互性运维分析**

| | |
|---|---|
| 用户 | 日志中出现如下命令该怎么办：net use \\192.168.xx.xx /del /y |
| 智慧体 | 如果您在日志中遇到了此命令，意味着有人试图删除与IP地址为192.168.xx.xx的计算机的网络连接。 |
| 用户 | 有什么建议? |
| 智慧体 | 请执行以下操作:<br><br>打开命令提示符或PowerShell窗口。可以通过按下Win + R组合键，键入"cmd"或"PowerShell"，然后按下回车键来打开它们。<br><br>在命令提示符或PowerShell窗口中，输入以下命令并按下回车键：net use \192.168.xx.xx /del /y |
| | 系统将尝试删除与该IP地址的网络连接。如果成功删除，您将收到一条确认消息。 |

运维智慧体愿景：并非数据驱动，而是指令驱动，可以进行根因查找与自我纠偏，**充当设备系统与工程师之间交流沟通的桥梁**

# PART 03

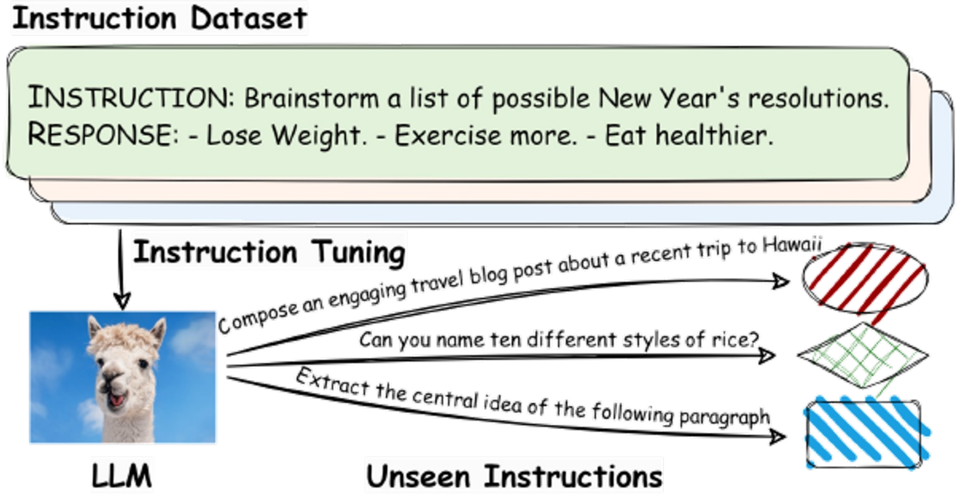
# 大模型Prompt引擎助力自适应运维智慧体：

## LogPrompt：利用Prompt工程激发大模型运维潜能，零样本推断+可解释性

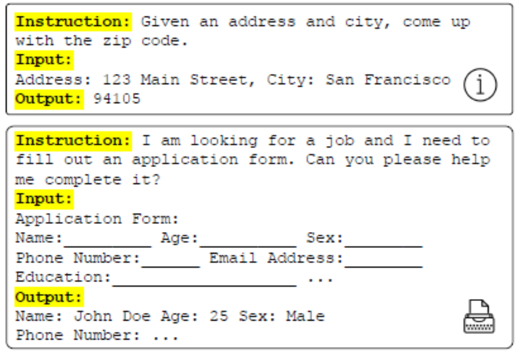
# LogPrompt解决传统日志分析两大Gap

| 传统方法 | | LogPrompt |
| --- | --- | --- |
| **依赖于任务数据，专家标注耗时耗力，自适应性差** | → | **无需训练资源，可灵活迁移至不同设备应用**<br>• 依托大模型预训练阶段内生通用知识，不再单独进行领域微调<br>• 基于Prompt策略注入领域专家对齐信息，快速灵活迁移 |
| **智慧有限，可解释性差，直接输出告警结论，无法实现告警事件分析** | → | **增强分析结果的可解释性、可交互性**<br>• 以思维链提示引擎激发大语言模型的领域文本分析能力和根因推理能力，在告警日志纷杂的信息中梳理思维链逻辑，AI模型端到端生成事件分析总结，快速判断漏报、误报，找出根因。<br>• 根据用户需求描述，以多轮对话的方式灵活地提供告警查询、定位、分析服务。 |

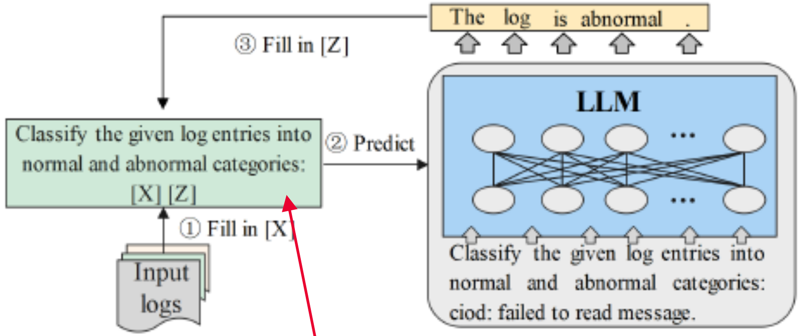
# LLM作为运维智慧体的潜力与挑战：大模型有强语言泛化与解释能力，但是对Prompt敏感




Large language models (LLMs) have powerful generalization ability to **unseen user instructions (Gap 1)**, and may also be able to handle unseen logs in the online situation of log analysis.



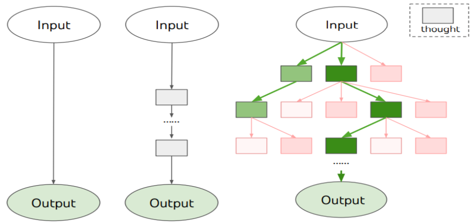

Unlike existing deep learning models, LLMs (such as ChatGPT) has strong language generating ability and can **handling complex writing tasks (Gap 2)** like email, report, etc. Log interpretation can be seen as a domain writing task.




| Methods[b] | BGL | | Spirit | |
|---|---|---|---|---|
| | S-F1[a] | T-F1 | S-F1 | T-F1 |
| ChatGPT (simple prompt) | 0.189 | 0.115 | 0.445 | 0.086 |
| +Implicit CoT | 0.307 | 0.183 | 0.443 | 0.096 |
| +Explicit CoT | **0.384** | **0.321** | **0.450** | **0.116** |

In our preliminary experiments, ChatGPT with the simple prompt achieved an F1-score of only 0.189 in anomaly detection. However, our best prompt strategy outperformed the simple prompt by 0.195 in F1-score.

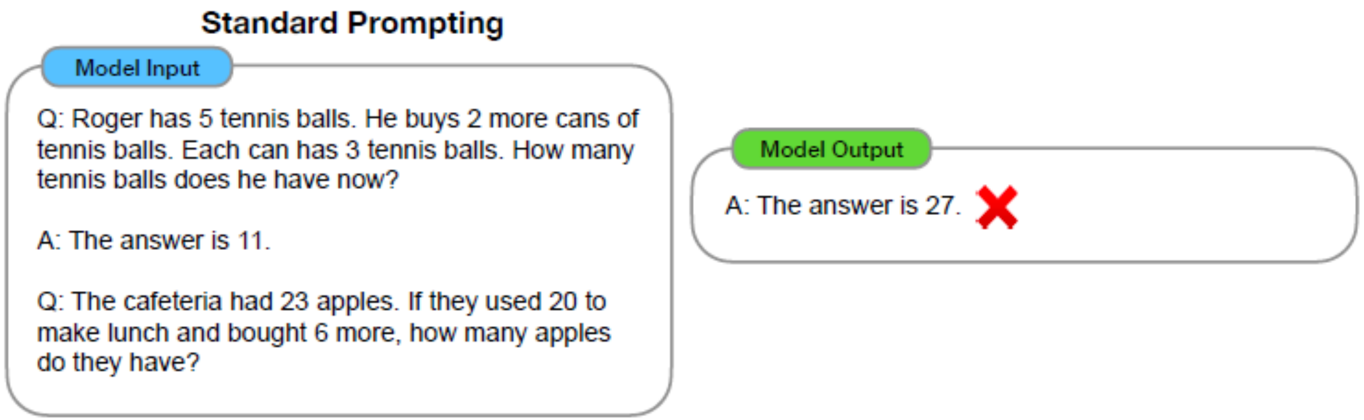
Since log analysis is a domain-specific and non-general NLP task, directly applying a simple prompt to LLMs can result in poor performance.



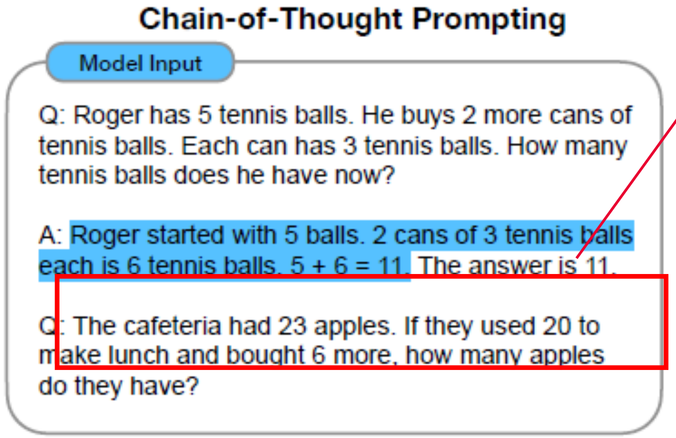

There are many prompt philosophies proposed in NLP tasks, such as CoT, ToT, etc.

The primary objective of LogPrompt is to enhance the correctness and interpretability of log analysis in the online scenario, through exploring a **proper strategy of prompting LLMs**.

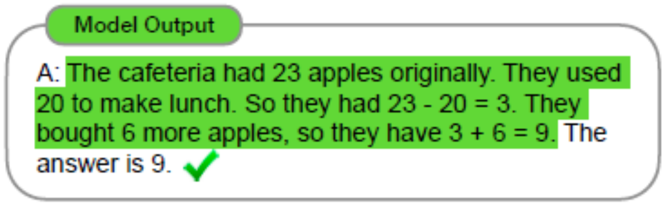
# 引入 chain-of-thought (CoT) prompt 策略可以激发LLM解决日志分析挑战的能力




The concept of chain of thought (CoT), a series of intermediate reasoning steps, is introduced by Wei et al.[1]. The CoT prompt **emulates the human thought process by requiring the model to include thinking steps** when addressing complex problems and can enhance the performance of LLMs in challenging tasks, such as solving mathematical problems.




The CoT Prompt in the original CoT paper put an example with intermediate steps before an input math problem. So that the model is encouraged to follow the thinking style in the example.




**Advantages of CoT prompt**

- Break down unseen problems into manageable steps **(Gap 1)**
- Enhance interpretability and transparency of LLM output **(Gap 2)**
- Unleash the learned abilities in the pre-training phase


[1] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le,D. Zhou et al., “Chain-of-thought prompting elicits reasoning in large language models,” Advances in Neural Information Processing Systems, vol. 35, pp. 24 824–24 837, 2022.

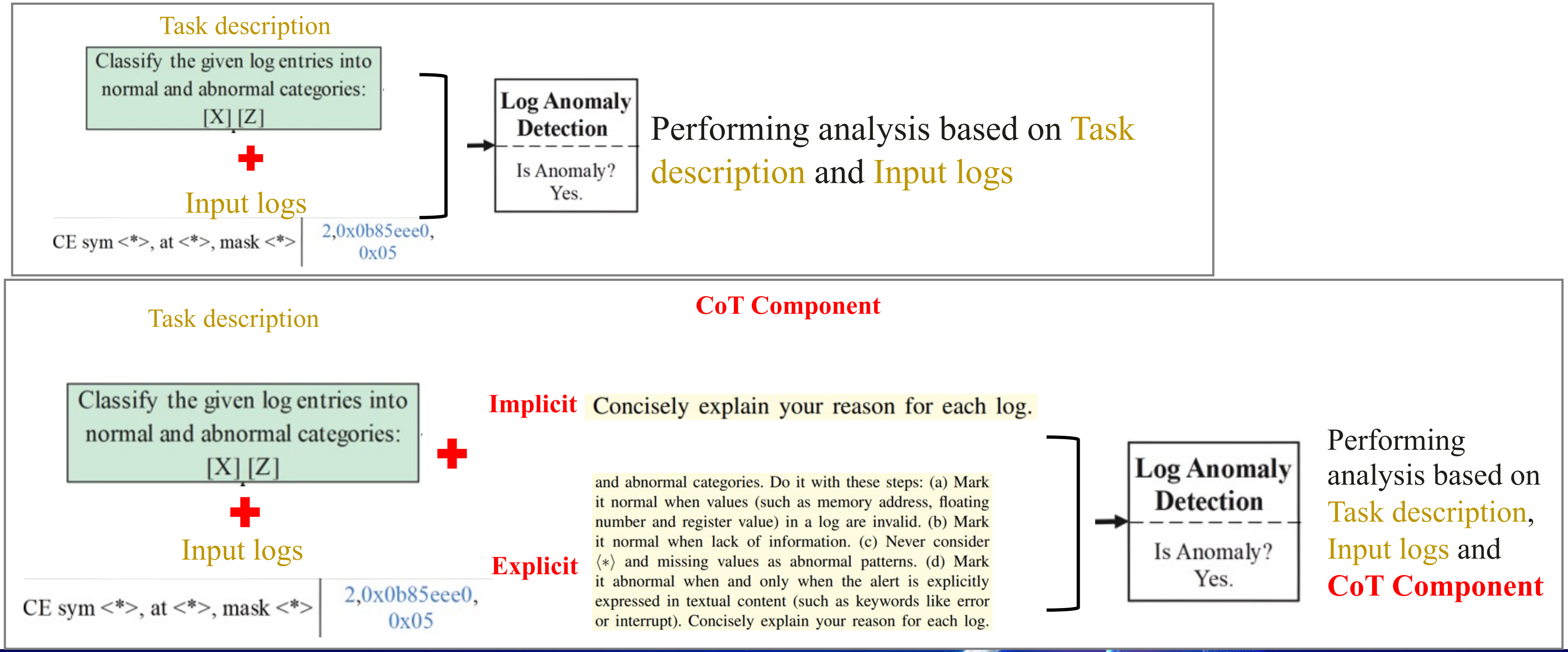
# LogPrompt 探索: 将 CoT prompt 的思想引入日志分析任务

In manual log analysis, practitioners also engage in a series of reasoning steps to reach a conclusion. For instance, without further definitions, the boundary between a normal log and an abnormal log is unclear. **To emulate the thinking process of O&M engineers**, we propose two variants of CoT prompt for log analysis:

- **Implicit CoT:** Human mostly has reasons before conclusion. Thus, in the prompt, we require the LLM to generate a reason for each normal/abnormal answer, justifying its decisions.
- **Explicit CoT:** We further explicitly define intermediate steps to regulate the thinking process. For example, in the task of anomaly detection, we constrain the definition of anomaly to be only "alerts explicitly expressed in textual content" and define four steps for anomaly detection.

Standard Prompt

LogPrompt (CoT)



# LogPrompt 探索: 其他可以在日志分析任务中应用的Prompt策略

- **Self-prompt:** this strategy involves the LLM suggesting its own prompts. A meta-prompt describing the task asks the LLM to generate prompt prefix candidates.

KE

The task of log parsing aims to extract common segments (template) and unique segments (variable) from raw logs. For example: raw log "visible is system.call.count gt 0" is parsed into a template "visible is * gt *". Now please specify several prompts that can make large language models precisely perform this task, where a list of input logs is parsed into their correspondent templates using * to replace the variables.

Here are some prompts that could be used to train a large language model to perform log parsing:

1. "Parse the following log into a template format, replacing variable parts with a wildcard (*): `[log]`"
2. "Convert the following log into a standardized template by identifying and replacing the variable parts with a *: `[log]`"
3. "Transform the raw log `[log]` into a log template by replacing variable segments with *"
4. "Given the following log, extract its common structure and replace variable parts with * to create a log template: `[log]`"
5. "Create a log template from the raw log `[log]` by replacing dynamic segments with *"

These candidates are then tested on a specific log dataset (in our case the first 100 logs from Android), with the most effective prompt chosen based on performance.

*Self-prompt*: $\arg\max_{p \in C}\{s(p)\}\ f_x([X])\ f_z([Z], S_p)$

- **In-context Prompt**: This approach uses several samples of labeled logs to set the context for the task. The LLM then predicts on new logs, using the context from the sample logs.

*In-context prompt*: Classify the given log entries into 0 and 1 categories based on semantic similarity to the following labelled example logs: (1) Log: $L_1$ Category: 1 (2) Log: $L_2$ Category: 0 ...(m-1) Log: $L_{m-1}$ Category: 1 (m) Log: $L_m$ Category: 0. $f_x([X])\ f_z([Z], S_c)$

- **Format Control:** We employ two functions, $f_X$ ([X]) and $f_Z$ ([Z], to establish the context for the input slot [X] and the answer slot [Z] in the prompt.

$f_x([X])$ = There are N logs, the logs begin: (1)$[X]_1$\n (2)$[X]_2$\n ...\n (N)$[X]_N$

$f_z([Z], S)$ = Organize your answer to be the following format: (1) $x$-$y$\n (2) $x$-$y$\n ...\n (N) $x$-$y$, where $x$ is $S$ and $y$ is the reason. [Z]

S is a text string describing the desired answer value range, like "a binary choice between abnormal and normal", or "a parsed log template".

# 实验设置

| Dataset | #Messages | #Templates (2k) | #Anomalies | L.P.[a] | A.D.[a] |
|---|---|---|---|---|---|
| HDFS | 82,293,702 | 14 | - | ✓ | - |
| Hadoop | 394,308 | 114 | - | ✓ | - |
| Zookeeper | 74,380 | 50 | - | ✓ | - |
| BGL | 4,713,493 | 120 | 348,460 | ✓ | ✓ |
| HPC | 433,489 | 46 | - | ✓ | - |
| Linux | 25,567 | 118 | - | ✓ | - |
| Proxifier | 21,329 | 8 | - | ✓ | - |
| Android | 30,348,042 | 166 | - | ✓ | - |
| Spirit | 7,958,733 | - | 768,142 | - | ✓ |

[a] **L.P.** denotes the task of log parsing, **A.D.** denotes anomaly detection.
✓ means the dataset is used as a task evaluation set with human labels.

- The effectiveness of LogPrompt is evaluated mainly using the logHub datasets,which contains **real-world logs from nine different domains**, including supercomputers, distributed systems, operating systems, mobile systems, and server applications.
- Two of the datasets (**BGL and Spirit**) were annotated by domain experts to identify **anomalous events** for the purpose of anomaly detection.
- To evaluate the log parsing performance, **eight of the datasets** have **log templates manually extracted** from a subset of 2000 log messages in each domain.
- All log data were **timestamped**, enabling the datasets to be split into train/test sets chronologically.

- In our primary experiments, the underlying LLM is accessed via APIs provided by external services.
- The initial temperature coefficient is set to 0.5, maintaining a balance by increasing the model's reasoning capabilities through diverse token exploration while limiting detrimental randomness.
- If the response format is invalid, the query is resubmitted with an increased temperature coefficient of 0.4 until the response format is correct. The format failure rate is less than 1%, which is consistent with existing literature.
- The train/test datasets are split chronologically to simulate online scenarios.

# 实验: LogPrompt在零样本日志分析场景取得了较好效果，因而可以减轻算法的数据依赖

- **Task of Log Parsing**

| Methods[a] | HDFS | Hd[b] | Zk | BGL | HPC | Linux | Px | Andro | Avg. |
|---|---|---|---|---|---|---|---|---|---|
| IPLoM [23] | 0.389 | 0.068 | 0.225 | 0.391 | 0.002 | 0.225 | 0.500 | 0.419 | 0.277 |
| LKE [8] | 0.424 | 0.198 | 0.225 | 0.379 | 0.381 | 0.388 | 0.309 | 0.000 | 0.288 |
| LogSig [34] | 0.344 | 0.050 | 0.225 | 0.333 | 0.002 | 0.146 | 0.339 | 0.116 | 0.194 |
| FT-tree [42] | 0.385 | 0.046 | 0.186 | 0.497 | 0.002 | 0.211 | 0.420 | 0.581 | 0.291 |
| Spell [6] | 0.000 | 0.058 | 0.045 | 0.536 | 0.000 | 0.091 | 0.000 | 0.245 | 0.122 |
| Drain [9] | 0.389 | 0.068 | 0.225 | 0.397 | 0.002 | 0.225 | 0.500 | 0.413 | 0.277 |
| MoLFI [26] | 0.000 | 0.095 | 0.000 | 0.333 | 0.000 | 0.026 | 0.000 | 0.208 | 0.083 |
| LogParse [24] | 0.632 | 0.502 | 0.348 | 0.665 | 0.330 | 0.588 | 0.334 | 0.233 | 0.454 |
| LogStamp [37] | 0.523 | 0.594 | 0.275 | 0.818 | 0.434 | 0.658 | 0.438 | **0.899** | 0.580 |
| LogPPT [19] | 0.838 | 0.526 | 0.795 | **0.982** | 0.287 | 0.423 | 0.638 | 0.313 | 0.600 |
| **LogPrompt** | **0.863** | **0.763** | **0.889** | 0.865 | **0.759** | **0.766** | **0.653** | 0.819 | **0.797** |

[a] LogPrompt uses **no training data**, unlike others trained with up to 10% logs.
[b] **Hd**, **Zk**, **Px** and **Andro** denotes Hadoop, Zookeeper, Proxifier and Android.

- For each dataset, most baseline methods are trained on the first 10% logs and evaluated on the remaining 90% logs, while LogPrompt is directly tested on the remaining 90% logs **without in-domain training data**.
- We adopt the F1-score as the metric. To calculate it, we tokenize the predicted log template into a list of tokens, then we treat the tokens as results from a classification task of {template, variable}.
- LogPrompt achieved the best F1-score on six of the eight datasets, and outperformed existing methods which require resources for in-domain training.

- **Task of Anomaly Detection**

| Methods | #Train[a] | BGL[c] P | BGL[c] R | BGL[c] F | Spirit P | Spirit R | Spirit F |
|---|---|---|---|---|---|---|---|
| DeepLog [7] | | 0.156 | **0.939** | 0.268 | 0.249 | 0.289 | 0.267 |
| LogAnomaly [25] | 4000 | 0.016 | 0.056 | 0.025 | 0.231 | 0.141 | 0.175 |
| LogRobust [43] | | 0.095 | 0.425 | 0.156 | 0.109 | 0.135 | 0.120 |
| **LogPrompt**[b] | **0** | **0.249** | 0.834 | **0.384** | **0.290** | **0.999** | **0.450** |

[a] **#Train** denotes the number of logs utilized for training.
**P** denotes Precision. **R** denotes Recall. **F1** denotes F1-score.
[b] LogPrompt here is constructed via the strategy of CoT-prompt.
[c] We automatically extracted 1766 templates from BGL; 1297 from Spirit.
The reported F1-score is session level. See template level in Table 7.

- For baselines, the first 4000 logs in each dataset are used for training. Both LogPrompt and the trained baselines are then tested on the remaining logs.
- We report the session-level F1-score of anomaly. A session is formed using fixed-window grouping with a length of 100 log templates.
- Despite the existing methods being trained on thousands of logs, LogPrompt still achieved strong performances in both datasets **without utilizing any in-domain training data, with an average improvement of 55.9% in terms of F1-score.**

The advantage of LogPrompt makes it a suitable choice for log analysis in online scenarios

# 实验: LogPrompt 生成了可信、详细、相关的分析过程，可以增强日志分析的解释性

- **A Novel Evaluating Task of Log Interpretation**

Table 4: Human Scoring Criteria for Evaluating the Interpretability of Large Language Models in Log Analysis. Score Ranking: 1 - Not Interpretable, 2 - Low Interpretable, 3 - Moderately Interpretable, 4 - Highly interpretable, 5 - Very Highly interpretable.

| Scores | Usefulness: Log Parsing | Usefulness: Anomaly Detection | Readability |
|---|---|---|---|
| 1 | No variable is extracted, or explanations of those variables are irrelevant. | No justification on the anomaly beyond a simple prediction label. | The text contains numerous unintelligible elements or grammatical mistakes. |
| 2 | Explanations are meaningless or logically wrong, hindering engineers from interpreting the logs. | The Justification for the prediction is irrelevant, or logically inconsistent with the facts. | Most of the generated text is readable, but it may have grammar errors or unclear phrases. |
| 3 | Variables with appropriate classes are partially extracted, and explanations are somewhat relevant. | The justification well supports the predictions, but may lack clarity and details. | The text has few grammar errors, although some terms may need refinement. |
| 4 | Variables with appropriate classes are mostly extracted, and explanations are specific and relevant. Engineers can understand the logs with less effort. | Specific, accurate, and relevant justification is presented, which positively assist engineers in eliminating false alarms and further analysis. | The text is clear, grammatically correct, with only a minimal number of technical terms possibly needing refinement. |
| 5 | Variables are fully and correctly extracted, and explanations are detailed, specific and relevant, enabling easy and precise understanding of logs. | Detailed, relevant and clear justification that significantly assists engineers in ruling out false alarms and locating the root cause. | The text is clear, detailed, grammatically perfect, and professional for software engineering. |

| Raters | Log Parsing | | | | Anomaly Detection | | | |
|---|---|---|---|---|---|---|---|---|
| | Usefulness | | Readability | | Usefulness | | Readability | |
| | Mean[a] | HIP[b] | Mean | HIP | Mean | HIP | Mean | HIP |
| R1 | 4.19 | 76.00% | 4.70 | 94.00% | 4.27 | 73.00% | 4.50 | 84.00% |
| R2 | 4.36 | 81.00% | 4.78 | 95.00% | 4.42 | 79.00% | 4.63 | 90.00% |
| R3 | 4.13 | 91.00% | 4.18 | 95.00% | 4.12 | 86.00% | 4.35 | 94.00% |
| R4 | 4.20 | 73.00% | 4.74 | 98.00% | 4.40 | 85.00% | 4.71 | 95.00% |
| R5 | 4.06 | 91.00% | 4.41 | 100.00% | 4.12 | 94.00% | 4.24 | 98.00% |
| R6 | 4.46 | 86.00% | 4.77 | 97.00% | 4.48 | 90.00% | 4.78 | 99.00% |
| Avg. | 4.23 | 83.00% | 4.60 | 96.50% | 4.30 | 84.50% | 4.54 | 93.33% |

[a] **Mean** is the average score of the ratings by a reviewer for all samples.
[b] **HIP** is the percentage of samples received a rating higher than four.

- **Usefulness.** The reviewers were asked to rate the extent to which the provided interpretation of the analysis decision is detailed, specific, relevant, logically sound and helpful in actual log analysis.
- **Readability.** The reviewers were asked to rate the ease with which a reader could understand the provided text. A text was considered readable if it was grammatically correct, meaningful, and professional.

- A total of **200 logs were randomly sampled** for the human evaluation, accompanied by LogPrompt's actual outputs, with 100 logs related to log parsing and 100 related to anomaly (evenly distributed across domains).
- Incorrectly predicted logs (FPs and FNs) were not included in this evaluation. An equal number of normal and abnormal samples were included for anomaly detection, and each selected log for log parsing was required to contain at least one variable.
- We invited **six practitioners to score the interpretability** level from 1 to 5 according to the criteria, independently across all the 200 samples **.**
- We reported two metrics: Mean and HIP. The average scores on the samples for both tasks in terms of usefulness and readability consistently exceeded four, and average HIP was consistently above 80%, indicating an overall helpful and readable content considered by experienced log analysts.

# 实验: LogPrompt 生成了可信、详细、相关的分析过程，可以增强日志分析的解释性

- **Feedbacks from Practitioners**

- "I appreciate the ability of LogPrompt in supporting the interpretation of logs from various domains. As our system continuously incorporating third-party services, sometimes I have **to read through the manuals to decipher logs from unfamiliar domains**. The explanations generated by **LogPrompt can provide a swift grasp of the logs** before I found the official definitions."

- "LogPrompt can definitely help **in the compose of analysis reports**, especially shortly after system crashes, where I often need to **quickly interpret the events** to non-technical colleagues in meetings."

- "In the realm of software O&M, **false alarms** are an inescapable reality, with false positives imposing substantial time costs on engineers and false negatives causing severe ramifications. Accompanying explanations with automatic analysis outcomes enables engineers to more **promptly ascertain the credibility of the purported anomaly**, thereby reducing the time spent on subsequent actions."

- **Bad Case Analysis**

| Cause | Ratio | Example | |
|---|---|---|---|
| | | Raw log | Interpretation |
| Domain Knowledge | 45.95% | CE sym 33, at 0x1ff2fc60, mask 0x04. | Sym 33 is a Object Name that refers to an object named "sym 33". |
| Semantic insufficiency | 40.54% | fpr8=0x3883a497 bfe016b5 8da3311e 3f06d5a7. | Normal - Register values. |
| Other | 13.51% | - | - |

- A major factor of bad cases is the LLM's lack of domain knowledge, which leads to overly general interpretations of some domain-specific terms. For example, it may refer to specific parameters as simply an object.

- Another cause is the lack of semantic contents in certain input logs, which can be attributed to their brevity or richness of non-NLP patterns (i.e., digits, codes and addresses).

# 三种Prompt策略的消融分析

## Self-Prompt

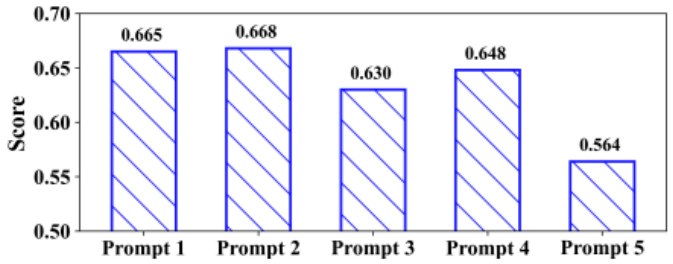
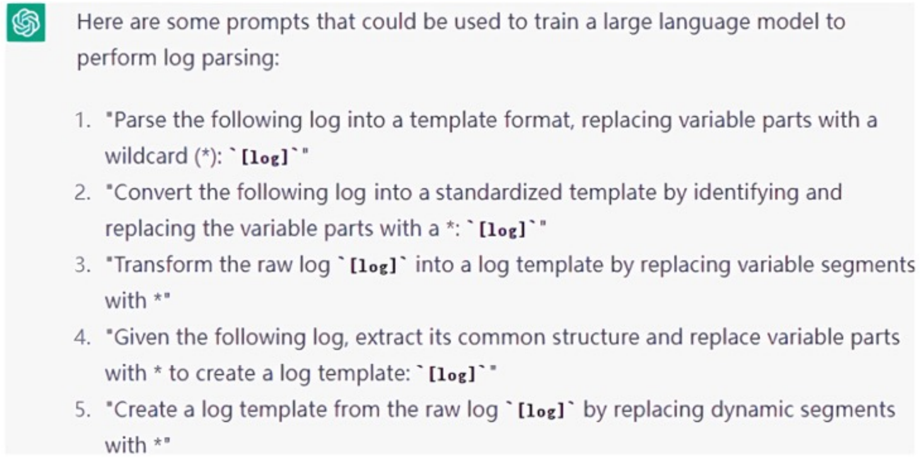



**Figure 5: Performance comparison of the five raised prompt candidates in Fig. 4 for Log Parsing.**

Compared to prompt 5, prompt 2 provides **more formal and accurate words** (such as "standardized" and "convert") and clearly **outlines the intermediate steps** for the task (identifying and replacing variables, then converting to a template).


Here are some prompts that could be used to train a large language model to perform log parsing:

1. "Parse the following log into a template format, replacing variable parts with a wildcard (*): `[log]`"
2. "Convert the following log into a standardized template by identifying and replacing the variable parts with a *: `[log]`"
3. "Transform the raw log `[log]` into a log template by replacing variable segments with *"
4. "Given the following log, extract its common structure and replace variable parts with * to create a log template: `[log]`"
5. "Create a log template from the raw log `[log]` by replacing dynamic segments with *"

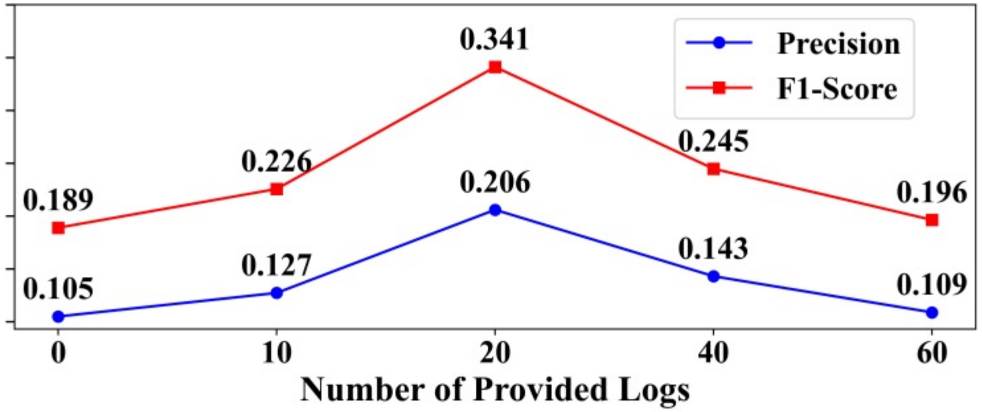
## CoT Prompt

| Methods[b] | BGL | | Spirit | |
|---|---|---|---|---|
| | S-F1[a] | T-F1 | S-F1 | T-F1 |
| ChatGPT (simple prompt) | 0.189 | 0.115 | 0.445 | 0.086 |
| +Implicit CoT | 0.307 | 0.183 | 0.443 | 0.096 |
| +Explicit CoT | **0.384** | **0.321** | **0.450** | **0.116** |

[a] **S-F1** denotes the session-level F1-score. **T-F1** denotes the template-level F1-score.
[b] The implicit CoT requires ChatGPT to report justifications. The explicit CoT further adds intermediate steps in prompts.

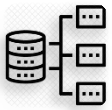
Interestingly, only **utilizing the implicit CoT** (requiring generating reasons) can still improve the model performance, likely due to the reason that with more NLP explanations, **the distribution of the generated answers is more close** to that in the pre-training phases of models.

## In-context Prompt




An overly long context prefixed to the prompt may cause LLMs to pay less attention to the new input logs, thereby deteriorating the task performance, which is why the peak is at 20 sample logs.

# 实践案例-算法实践：xx云基于大模型Prompt引擎实现本地API联通的定制化告警分析设备



**现网设备日志**

## Prompt引擎

中央控制器


策略控制器

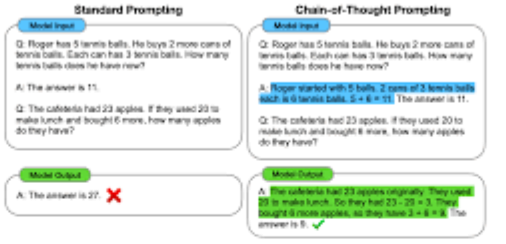
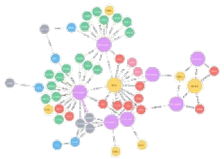
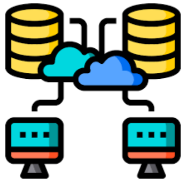


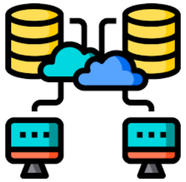

Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.



知识引擎

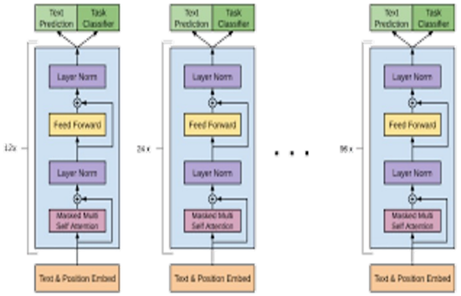


**现网数据库**

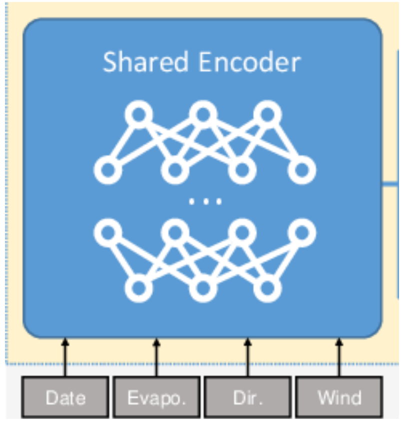


**离线知识库**

**任意大模型**



**大模型盘古**

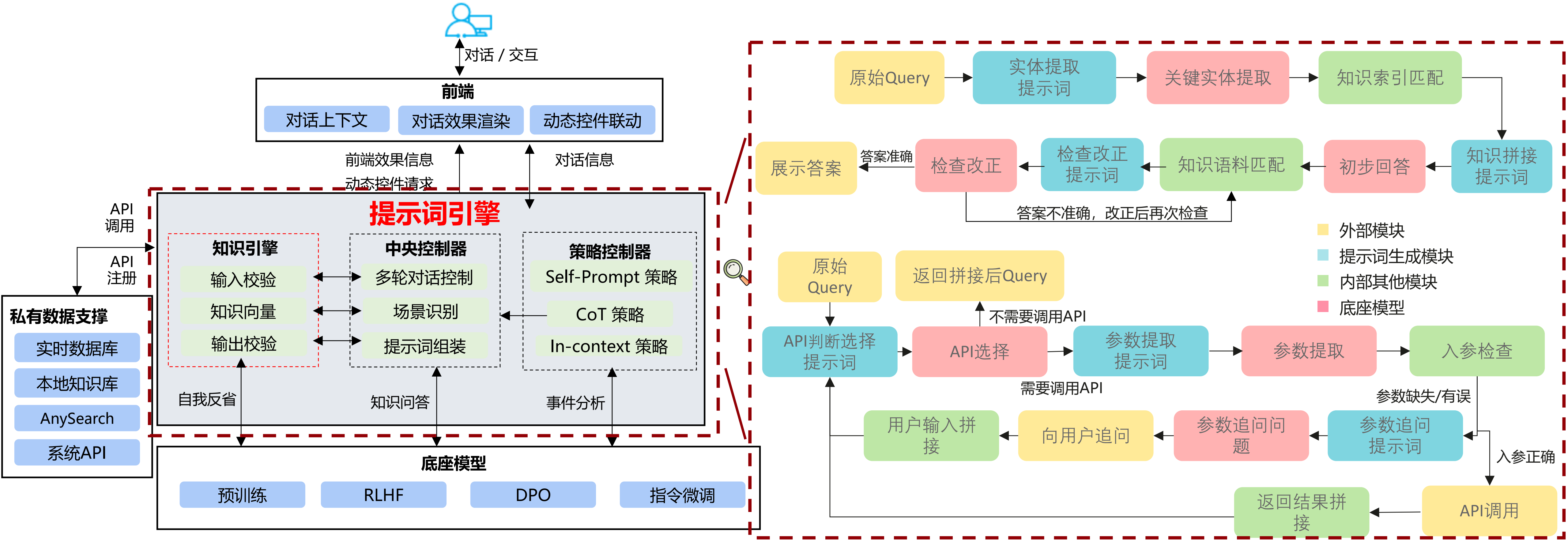



**开源大模型**

| | |
|---|---|
| 用户 | 日志中出现如下命令该怎么办：net use \\\\192.168.xx.xx /del /y |
| 智慧体 | 如果您在日志中遇到了此命令，意味着有人试图删除与IP地址为192.168.xx.xx的计算机的网络连接。 |
| 用户 | 有什么建议? |
| 智慧体 | 请执行以下操作：<br>打开命令提示符或PowerShell窗口。可以通过按下Win + R组合键，键入"cmd"或"PowerShell"，然后按下回车键来打开它们。<br>在命令提示符或PowerShell窗口中，输入以下命令并按下回车键：net use \192.168.xx.xx /del /y<br>系统将尝试删除与该IP地址的网络连接。如果成功删除，您将收到一条确认消息。 |

根据现网数据库分析告警事件，结合离线知识库进行业务问答。

# 实践案例-服务实践：Prompt引擎服务设计架构




系统架构（外部）

系统架构（内部）

# PART 04

# 大模型知识迁移打造运维专精模型：

## 工业场景需要增强可离线部署的小型LLM的日志分析能力，以更好地理解Prompt

# Beyond LogPrompt: 应用Prompt application策略到可部署的小型LLM存在领域知识Gap

## Applying LogPrompt to a smaller-scaled LLM: Vicuna 13B

- The deployment of large, proprietary, and API-dependent LLMs in local environments can be a challenging task.
- The reliability of services may be compromised if the API services of LLMs become unavailable.
- Therefore, for industrial usage, it is crucial for LogPrompt to have compatibility with alternative open-source, privacy-protected, and smaller-scale LLMs.

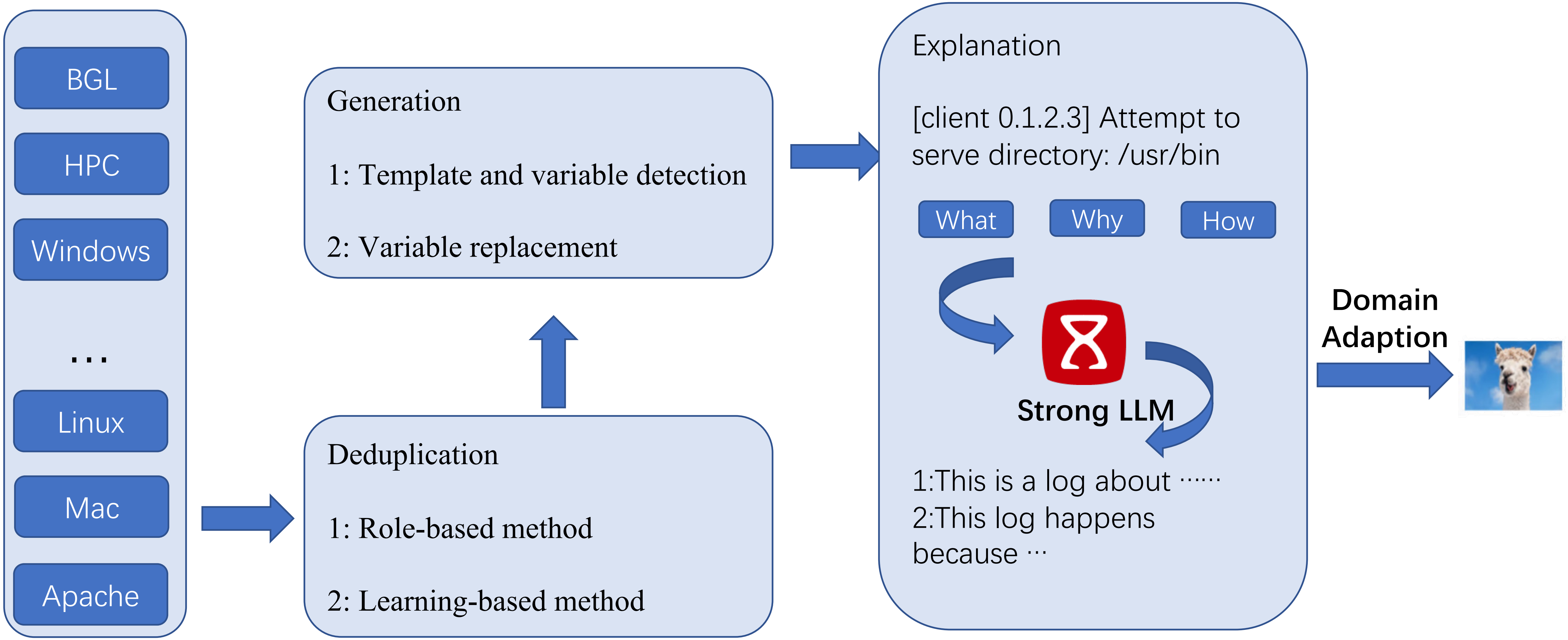
| LLMs | Size | HDFS | Hd[a] | Zk | BGL | HPC | Linux | Px | Andro |
|---|---|---|---|---|---|---|---|---|---|
| **GPT Model** | | | | | | | | | |
| *ChatGPT* | 175B | 0.863 | 0.763 | 0.889 | 0.865 | 0.759 | 0.766 | 0.653 | 0.819 |
| **Vicuna**[a] (with) | | | | | | | | | |
| *Simple Prompt* | 13B | 0.244 | 0.018 | 0.273 | 0.048 | 0.022 | 0.109 | 0.014 | 0.041 |
| *LogPrompt* | 13B | 0.779 | 0.468 | 0.364 | 0.266 | 0.243 | 0.715 | 0.614 | 0.248 |

[a] **Hd**, **Zk**, **Px**, **Andro** denotes Hadoop, Zookeeper, Proxifier, Android.
[b] Vicuna with LogPrompt uses the in-context prompt strategy with $m$=3.

Online Log Parsing with smaller-scaled LLMs using LogPrompt

- Although Vicuna has only 13B parameters, when equipped with LogPrompt, it achieves a comparable performance of log parsing with the 175B GPT model in the datasets of HDFS, Linux and Proxifier.
- Additionally, when the prompt strategy transitions from a simple prompt to LogPrompt, Vicuna exhibits significant improvements on performance, with an average increase of 380.7% in F1-score.
- As Vicuna is open-source and requires fewer resources for training and deployment, the success of LogPrompt on Vicuna holds promising implications for building in-domain industrial applications.
- **The performance of smaller-scaled LLMs like Vicuna still has a room for improvement. Since the base model is only 13B, it might not fully understand the domain-specific content in actual logs. Domain adaption techniques may help these models to better following the strategies of LogPrompt.**

# SuperLog: 对开源小型LLM进行领域知识注入，以增强其对领域Prompt策略的理解执行能力



# SuperLog: 初步实验结果

- 任务：日志解析
  - 实验设置：10%训练，90%推理
  - Baseline: LLaMA-7B直接下游任务微调
  - **指标：F1-Score**

| | HDFS | Hadoop | Zookeeper | Linux | Proxifier |
|---|---|---|---|---|---|
| LLaMA-7B | 0.9997 | 0.937 | 0.786 | 0.914 | 0.929 |
| Superlog | 0.988 | **0.942** | **0.815** | 0.914 | **0.939** |

  - **指标: RandIndex**

| | HDFS | Hadoop | Zookeeper | Linux | Proxifier |
|---|---|---|---|---|---|
| LLaMA-7B | 0.949 | 0.969 | 0.999 | 0.960 | 0.898 |
| Superlog | **0.979** | **0.982** | 0.998 | **1.000** | **0.998** |

  - **指标: Adjusted-RandIndex**

| | HDFS | Hadoop | Zookeeper | Linux | Proxifier |
|---|---|---|---|---|---|
| LLaMA-7B | 0.724 | 0.866 | 0.993 | 0.895 | 0.788 |
| Superlog | **0.897** | **0.925** | 0.989 | **1.000** | **0.995** |

# PART 05

## 未来畅想

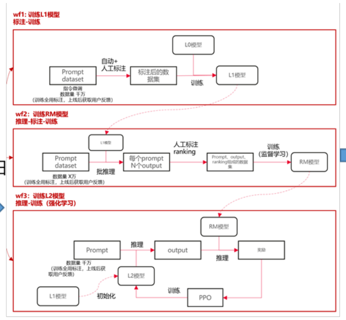
# AI运维智慧体面临更高要求：文本、图片、语音全模态支持

从应用角度来看，从**文本**、**图片**、**语音视频**等全模态支撑LLMOps运维系统。

从构建方式来看，自动抓取系统流量、截图、客服记录等进行自监督学习。

日志流量


系统历史数据：
{告警，案例，事件，日志，调用链}



强化学习流水线

运维智慧体

L1：目标自驱动辅助决策大模型

故障定位/报告生成/流量分析/根因查找/咨询/建议/方案推荐。。。

原子化复杂指令

结合领域知识进行指令微调
泛化未见过的新指令

L0：运维指令大模型

领域知识库

领域自适应

**文本**

模型：GPT3.5/4
LLaMA/Qwen

**Prompt：**以上日志怎么理解？ —— 日志解析

**Prompt：**系统是否有异常？ —— 异常检测

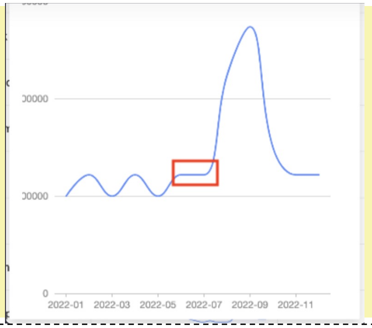
**Prompt：**帮我分析一下根因 —— 根因分析

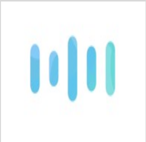
**Prompt：**把上述内容做个摘要报告 —— 报告生成

**图片**

模型：StableDiffusion
CogVLM/LLaVA

**Prompt:** 根据该图分析系统流量变化趋势?
**Response:**该图片描述了项目&文档&开发视图页面每月的使用趋势。其中横坐标为一天中的各个时间，纵坐标表示项目&文档&开发视图页面使用数量，单位为个，总体趋势在9月份有高峰。

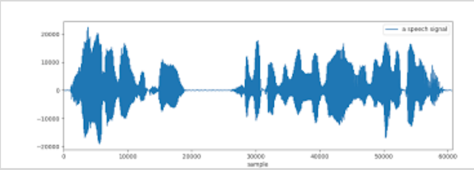


**语音**

模型：Whisper
GPT-4o

**Prompt：**帮我打开控制台目标页面，并告诉我最近的事件发生地址以及相关解决方案。



# THANKS