腾讯云可观测平台

LLM 可观测

腾讯云

文档目录

LLM 可观测
LLM 可观测简介

最近更新时间：2025-11-28 11:09:22

LLM 可观测是专为 LLM（大语言模型）应用量身定制的应用性能管理平台，基于腾讯云应用性能监控（Application Performance Management，APM）技术底座构建。LLM 可观测能力打破了传统监控对 AI 场景的适配局限，深度贴合 LLM 应用的技术特性与运行逻辑，不仅能覆盖传统应用的性能监控需求，更聚焦 LLM 特有的模型调用、响应生成等核心环节。LLM 可观测提供轻量级无侵入式接入方案，无需改动业务代码，即可快速开启，帮助开发者实时掌握应用运行状态，精准排查问题，为 LLM 应用的稳定、高效运行提供全方位保障。

计费规则

LLM 可观测的计费规则与腾讯云应用性能监控（Application Performance Management，APM）一致，无额外计费项，详情请参见 应用性能监控计费概述 。

LLM 可观测优势

- **LLM 场景专属适配**：针对 Prompt 处理、LLM 工作流、模型调用等核心环节定制观测能力，深度贴合 LLM 应用的技术特性与运行逻辑。重点体现响应延迟、调用成功率、Token 使用、模型质量等 LLM 关键指标，直击核心需求。
- **全链路数据打通**：全链路覆盖**业务层**，**调用层**和**模型服务层**，清晰呈现请求流转与性能瓶颈。
- **快速接入**：针对 Python、Go 等语言，提供轻量级无侵入式接入方案，无需改动业务代码，即可快速启用 LLM 可观测能力。同时，支持对常见 **LLM 框架**和**传统（非 LLM）框架**的自动埋点，兼容 OpenTelemetry 协议标准，能够和其他使用 OpenTelemetry 方案接入的应用实现链路信息互通。
- **问题高效定位**：优化 AI 场景下的告警策略与下钻分析功能，快速定位根因，提升排查效率。

快速入门

请参见 快速入门 完成授权、资源创建以及 LLM 应用的接入。

接入 LLM 应用
接入 Python LLM 应用
通过 OpenTelemetry-Python 探针接入

最近更新时间：2025-12-05 15:19:52

本文将通过相关操作介绍如何通过腾讯云 OpenTelemetry-Python 探针接入 Python LLM 应用。

支持的 LLM 组件与框架

腾讯云 OpenTelemetry-Python 探针基于社区 OpenTelemetry Python 项目二次开发，同时支持对**常见 LLM 框架和传统（非 LLM）Python 框架**的自动埋点，兼容 OpenTelemetry 协议标准，能够和其他使用 OpenTelemetry 方案接入的应用实现链路信息互通。腾讯云 OpenTelemetry-Python 探针支持以下组件或框架：

支持的组件与框架	链接
LLM 组件与框架	• OpenAI SDK（openai >= 0.27.0）：OpenAI 官方提供的 API 封装，用于直接调用所有兼容 OpenAI 标准的大模型。 • Ollama（ollama >= 0.4.0）：本地运行和管理开源大模型的轻量化推理框架。 • LangChain / LangGraph（langchain-core > 0.1.0）：用于构建和编排大模型应用的工作流框架，支持复杂链路与状态管理。 • LlamaIndex（llama-index >= 0.7.0，llama-index-core >= 0.7.0）：专注于把外部数据接入 LLM 的 RAG 框架，提供检索与索引能力。
传统 Python 组件与框架	完整的 支持列表 。

> **说明：**
> 该方案支持 **Python 3.9及以上版本**。

接入流程

获取接入点与 token

1. 登录 腾讯云可观测平台 控制台。
2. 在左侧菜单栏中选择 **LLM 可观测**，单击**应用列表 > 接入应用**。
3. 选择您所要接入的**地域**以及**业务系统**。

4. 选择您想要的**上报方式**，获取您的**接入点**和 **Token**。

> **说明：**
> - 内网上报：使用此上报方式，您的服务需运行在腾讯云 VPC。通过 VPC 直接连通，在避免外网通信的安全风险的同时，可以节省上报流量开销。
> - 外网上报：当您的服务部署在本地或非腾讯云 VPC 内，可以通过此方式上报数据。请注意外网通信存在安全风险，同时也会造成一定上报流量费用。

安装 pip 包

通过 `pip` 命令安装腾讯云自研探针，其中包含 OpenTelemetry-SDK 的相关依赖。

```
pip install tapm-distro opentelemetry-exporter-otlp==1.34.1

tapm-bootstrap -a install
```

命令行方式上报

加上 tapm-instrument 前缀完成埋点和启动，假设原来的项目启动命令是 `python app.py`，现在可以通过如下命令启动 Python 应用。

```
tapm-instrument --traces_exporter otlp \
--metrics_exporter otlp \
--logs_exporter none \
--service_name <service_name> \
--resource_attributes "token=<token>,host.name=<host.name>" \
--exporter_otlp_endpoint <endpoint> \
python app.py
```

对应的字段说明如下，请根据实际情况进行替换。

- `<service_name>`：应用名，多个使用相同 serviceName 接入的应用进程，会表现为相同应用下的多个实例。应用名最长63个字符，只能包含小写字母、数字及分隔符“-”，且必须以小写字母开头，数字或小写字母结尾。
- `<token>`：前置步骤中拿到业务系统 Token。
- `<host.name>`：该实例的主机名，是应用实例的唯一标识，通常情况下可以设置为应用实例的 IP 地址。
- `<endpoint>`：前置步骤中拿到的接入点。

接入验证

完成接入工作后，启动 LLM 应用，在 LLM 可观测 > 应用列表 页面将展示接入的应用。由于可观测数据的处理存在一定延时，如果接入后在控制台没有查询到应用或实例，请等待30秒左右。
同时，在 LLM 可观测 > 链路追踪 中，也能够查询到相关的 Span 记录，通过单击第一列 traceID 对应的链接，可以进入链路详情视图分析链路的每个环节。

单击其中的 Span，能够获取更多详细信息，例如下图：

通过 Langfuse SDK 接入

最近更新时间：2025-12-05 15:26:01

Langfuse 是一个开源的 LLM 工程化平台，专为监控、调试和优化 LLM 应用而设计。它提供全面的追踪监控、提示词版本管理、成本分析、质量评估和数据集管理等核心功能，支持 LangChain、OpenAI SDK 等主流框架的无缝集成。通过直观的仪表板和详细的分析报告，Langfuse 帮助开发团队深入了解应用性能，控制运营成本，持续改进模型输出质量。
LLM 可观测已兼容 Langfuse 探针 协议，如果您之前使用 Langfuse 作为可观测平台，可以参考本文无缝切换到腾讯云 LLM 可观测。

前提条件

该方案支持 Langfuse SDK 3.2.6 及以上版本。

操作步骤

步骤1：配置数据发送目标

在 LLM 应用中，一般通过如下配置将监控数据发送到 Langfuse 平台：

```
# Langfuse authentication
os.environ["LANGFUSE_SECRET_KEY"] = "sk-lf-xxx"
os.environ["LANGFUSE_PUBLIC_KEY"] = "pk-lf-xxx"
os.environ["LANGFUSE_HOST"] = "https://cloud.langfuse.com"


from langfuse import Langfuse

langfuse_client = Langfuse()

# ----- 业务代码开始 -----

# ----- 业务代码结束 -----

# 保证退出前数据发送完
langfuse_client.flush()
```

在此基础上，只需添加如下代码，就能实现将 Langfuse 探针埋点产生的链路数据发送到腾讯云 LLM 可观测。

```
from opentelemetry import trace
```

```
from opentelemetry.sdk.trace import TracerProvider
from opentelemetry.exporter.otlp.proto.grpc.trace_exporter import
OTLPSpanExporter
from opentelemetry.sdk.trace.export import BatchSpanProcessor
from opentelemetry.sdk.resources import Resource

# ----- Langfuse 平台相关配置（可移除） -----
os.environ["LANGFUSE_SECRET_KEY"] = "sk-lf-xxx"
os.environ["LANGFUSE_PUBLIC_KEY"] = "pk-lf-xxx"
os.environ["LANGFUSE_HOST"] = "https://cloud.langfuse.com"

# ----- LLM 可观测相关配置 -----
resource = Resource.create({
    "service.name": "<serviceName>", # 应用名，可自行配置
    "token": "<token>", # 接入 token，LLM 可观测控制台获取
    "host.name": "hostName", # 主机名，可自行配置
})
tracer_provider = TracerProvider(resource=resource)
tracer_provider.add_span_processor(
    BatchSpanProcessor(
        OTLPSpanExporter(
            endpoint="<endpoint>", # 接入点，LLM 可观测获取
        )
    )
)
trace.set_tracer_provider(tracer_provider)

from langfuse import Langfuse

langfuse_client = Langfuse(
    # 指定 Trace Provider
    tracer_provider=tracer_provider,
    # 阻止 Langfuse 埋点产生的链路数据发送到 Langfuse 平台。
    # 若不设置此参数，链路数据将同时发送到 Langfuse 平台和和腾讯云 LLM 可观测。
    blocked_instrumentation_scopes=["langfuse-sdk"]
)


# ----- 业务代码开始 -----
```

```
# ----- 业务代码结束 -----

# 保证退出前数据发送完
langfuse_client.flush()
```

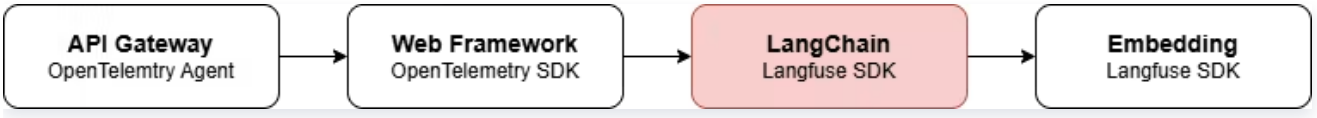
对应的字段说明如下：

- `<serviceName>`：应用名，多个使用相同 serviceName 接入的应用进程，会表现为相同应用下的多个实例。应用名最长63个字符，只能包含小写字母、数字及分隔符“-”，且必须以小写字母开头，以数字或小写字母结尾。
- `<token>`：前置步骤中拿到业务系统 Token。
- `<hostName>`：该实例的主机名，是应用实例的唯一标识，通常情况下可以设置为应用实例的 IP 地址。
- `<endpoint>`：前置步骤中拿到的接入点。

至此已完成将监控数据发送到腾讯云 LLM 可观测的配置。

步骤2：在链路埋点中标记 LLM 入口（可选）

LLM 可观测引入了 LLM 入口概念，代表一条链路中开启 LLM 调用的环节。如果您的分布式调用链路**在 Langfuse SDK 链路埋点之前，链路的上游使用了其他 OpenTelemetry 方案（例如腾讯云 OpenTelemetry-Python 探针）**，则需要在链路埋点标记 LLM 入口，否则将影响 LLM 可观测的数据展示。



标记方式

根据开启 LLM 调用环节的实际情况，您可以基于 OpenTelemetry SDK 或 Langfuse SDK 标记 LLM 入口，标记方式为：在当前 Span 添加 Span 属性，将 `gen_ai.is_entry` 设置为 `True`。

通过 OpenTelemetry SDK 进行标记

通常标记于 LLM 调用的发起环节，例如通过 OpenAI 访问大模型。

```
@observe(name="run_demo")
def run_demo() -> None:
    question = "什么是 OpenTelemetry? "
    print(f"User: {question}")

    # 在此处添加入口标记
    trace.get_current_span().set_attribute("gen_ai.is_entry", True) #
gen_ai.is_entry: Bool(true)
```

```
    answer = chat_completion_stream(question)
    print("\n\nAssistant (final):\n" + answer)
```

通过 Langfuse SDK 添加标记

以 LangChain 和 LangGraph 为例，通常标记在组件的执行入口（invoke）之前。

```
with langfuse.start_as_current_span(name="langgraph-agent-demo") as
span:
    span.update(metadata={"gen_ai.is_entry": True}) # 在此处添加，只会在入口
span 处标记
    # 注意不能通过以下方式添加在 metadata 中，这样会使所有的后继 span 都继承这个属
性，让入口标记失去意义
    result = agent_executor.invoke(
        {"input": question},
        config={
            "callbacks": [callback_handler],
            "metadata": {
              "langfuse_user_id": "user-456",
            }
        }
    )
```

Dify 应用接入

最近更新时间：2025-12-05 15:33:32

Dify 是一款开源的 LLM 应用开发平台，通过可视化的低代码方式，帮助用户快速构建、部署和管理生产级的 AI 应用、智能体（Agent）及 RAG 系统。腾讯云 LLM 可观测已集成到 Dify 自带的监控平台中，方便用户一键接入。

前提条件

Dify 版本 >= v1.10.0。

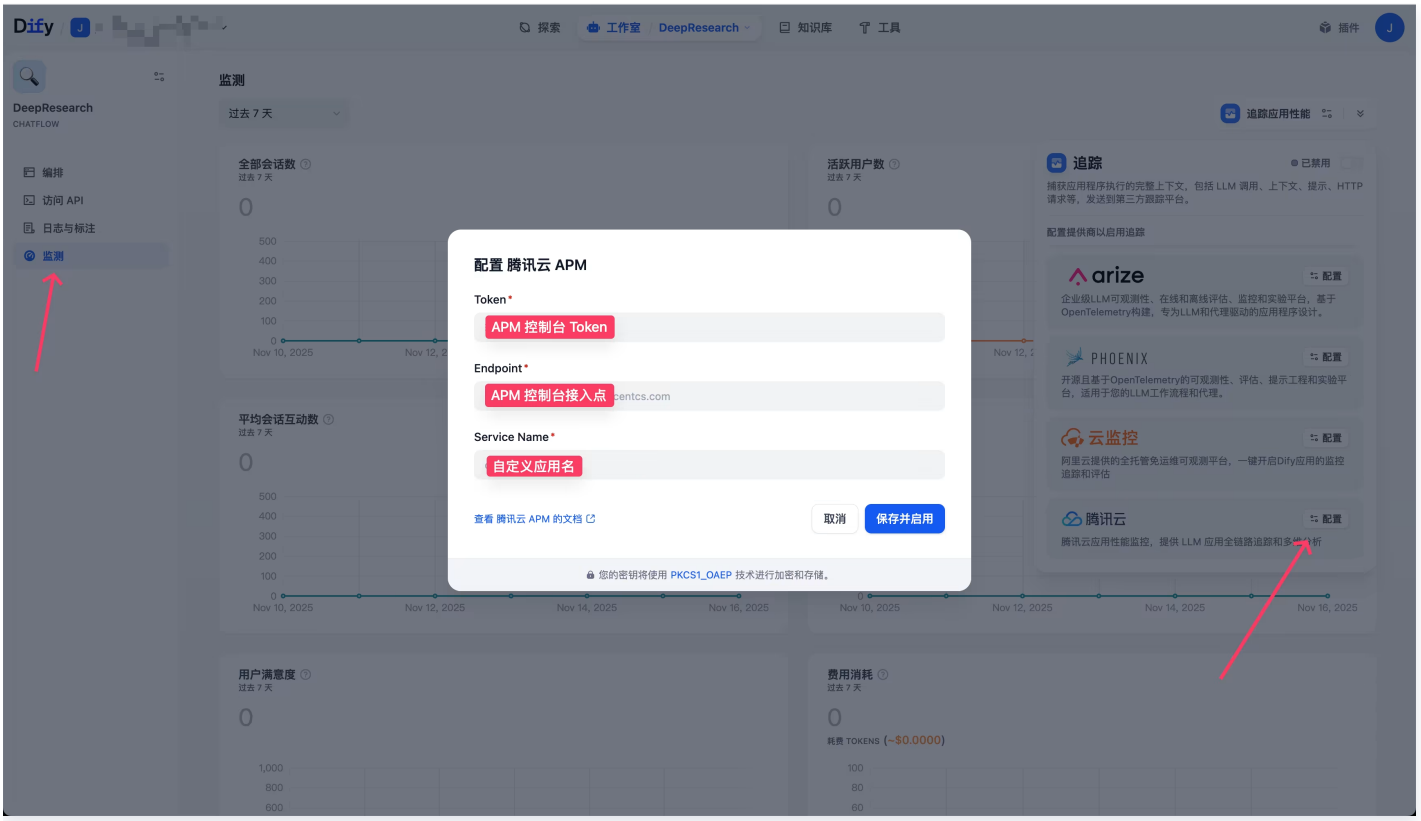
获取接入点与 token

1. 登录 腾讯云可观测平台 控制台。
2. 进入 LLM 可观测 > 应用列表，单击**接入应用**。
3. 选择您所要接入的**地域**以及**业务系统**。
4. 选择您想要的**上报方式**，获取您的**接入点**和 **Token**。

> **说明：**
> - 内网上报：使用此上报方式，您的服务需运行在腾讯云 VPC。通过 VPC 直接连通，在避免外网通信的安全风险的同时，可以节省上报流量开销。
> - 外网上报：当您的服务部署在本地或非腾讯云 VPC 内，可以通过此方式上报数据。请注意外网通信存在安全风险，同时也会造成一定上报流量费用。

在 Dify 平台配置

1. 在**工作室**页面，单击您需要接入的应用，进入应用详情页。
2. 进入左边栏的**监测**页面，选择右上角**追踪应用性能**，选择**腾讯云**。
3. 填入上一步获取到的 **Token** 和**接入点**。
4. 填入自定义应用名，单击**保存并启用**。多个使用相同应用名接入的应用进程，会表现为该应用下的多个实例。应用名最长63个字符，只能包含小写字母、数字及分隔符“-”，且必须以小写字母开头，数字或小写字母结尾。



接入验证

完成接入工作后，启动 LLM 应用，在 LLM 可观测 > 应用列表 页面将展示接入的应用。由于可观测数据的处理存在一定延时，如果接入后在控制台没有查询到应用或实例，请等待30秒左右。

接入 Go LLM 应用
tRPC-Agent-Go 接入

最近更新时间：2025-12-05 15:26:01

tRPC-Agent-Go 是腾讯开源的 Go 语言 AI Agent 开发框架，支持智能推理、持久化 memory、multi-agent 等功能，深度集成腾讯 tRPC 微服务生态，与已有的 tRPC-A2A-Go（Agent-to-Agent 通信框架）和 tRPC-MCP-Go（MCP 开发框架）形成完整的 Go 语言 AI 生态闭环。

操作步骤

trpc-agent-go 框架已经 内置了可观测功能，因此可以在不接入第三方探针的情况下，直接通过配置开启链路和指标的上报功能。

步骤1：获取接入点和 Token

1. 登录 腾讯云可观测平台 控制台。
2. 在左侧菜单栏中选择 LLM 可观测 > 应用列表，单击**接入应用**。
3. 在右侧弹出的**接入应用**抽屉框中，单击 Go 语言。
4. 在**接入 Go 应用**页面，选择您所要接入的**地域**以及**业务系统**。
5. 选择**接入协议类型**为 OpenTelemetry。
6. 选择您所想要的**上报方式**，获取您的**接入点**和 Token。

> **说明：**
> - 内网上报：使用此上报方式，您的服务需运行在腾讯云 VPC。通过 VPC 直接连通，在避免外网通信的安全风险的同时，可以节省上报流量开销。
> - 外网上报：当您的服务部署在本地或非腾讯云 VPC 内，可以通过此方式上报数据。请注意外网通信存在安全风险，同时也会造成一定上报流量费用。

步骤2：通过 trpc-agent-go 内置的可观测功能配置接入

> **注意：**
> tRPC-Agent-Go 本地采集的指标会上报到 APM，这里需要设置环境变量保证指标的聚合时间性：
> OTEL_EXPORTER_OTLP_METRICS_TEMPORALITY_PREFERENCE = DELTA

示例代码如下。

```
import (
```

```
    // 引入 OpenTelemetry SDK 相关包
    "go.opentelemetry.io/otel/attribute"

    // 引入 trpc-agent-go 内置的可观测功能包
    ametric "trpc.group/trpc-go/trpc-agent-go/telemetry/metric"
    atrace "trpc.group/trpc-go/trpc-agent-go/telemetry/trace"
)



func main() {
    // 业务相关代码

    // trpc-agent-go 配置 trace 和 metric 初始化
    ctx := context.Background()

    // Initialize OpenTelemetry tracing.
    cleanTrace, err := atrace.Start(
        ctx,
        atrace.WithEndpoint("<endpoint>"),
        atrace.WithServiceName("<service_name>"),
        atrace.WithResourceAttributes(attribute.String("token", "
<token>")),
    )
    if err != nil {
        log.Fatalf("Failed to start trace telemetry: %v", err)
    }
    defer func() {
        if err := cleanTrace(); err != nil {
            log.Printf("Failed to clean up trace telemetry: %v", err)
        }
    }()

    // Initialize OpenTelemetry metrics.
    mp, err := ametric.NewMeterProvider(
        ctx,
        ametric.WithEndpoint("<endpoint>"),
        ametric.WithServiceName("<service_name>"),
        ametric.WithResourceAttributes(attribute.String("token", "
<token>")),
```

```
    )
    if err != nil {
        log.Fatalf("Failed to create metric provider: %v", err)
    }
    defer func() {
        if err := mp.Shutdown(ctx); err != nil {
            log.Printf("Failed to clean up metric telemetry: %v", err)
        }
    }()

    if err := ametric.InitMeterProvider(mp); err != nil {
        log.Fatalf("Failed to init metric telemetry: %v", err)
    }

    // 业务相关代码
}
```

对应的字段说明如下，请根据实际情况进行替换。

- `<service_name>` ：应用名，多个使用相同 service name 接入的应用进程，会表现为该应用下的多个实例。应用名最长63个字符，只能包含小写字母、数字及分隔符“-”，且必须以小写字母开头，数字或小写字母结尾。
- `<token>` ：前置步骤中拿到业务系统 Token。
- `<endpoint>` ：前置步骤中拿到的接入点。

接入验证

完成接入工作后，启动 LLM 应用，在 LLM 可观测 > 应用列表 页面将展示接入的应用。由于可观测数据的处理存在一定延时，如果接入后在控制台没有查询到应用或实例，请等待30秒左右。

控制台操作指南
应用列表

最近更新时间：2025-11-28 11:09:22

应用列表功能用于查询接入 LLM 可观测的应用，是前往应用级监控分析的入口。

操作步骤

1. 登录 腾讯云可观测平台 。
2. 在左侧菜单栏中选择 LLM 可观测 > 应用列表 。
3. 通过右上方的时间选择器指定查询时间跨度，通过**应用过滤**对话框可以指定更多查询条件。

功能说明

- **应用过滤**：可以基于应用名、应用 ID 和应用标签进行过滤。
- **清理应用**：清理应用功能仅针对已经不再上报监控数据的应用，请先确保该应用的探针已经成功卸载（如果通过代码上报，请删除相关逻辑）。在执行完清理操作后，系统将删除和该应用相关的所有数据。
- **编辑标签**：通过该功能，可以为应用关联多个标签，用于应用查询以及细粒度的权限配置。标签键和标签值引用自腾讯云统一的标签中心，如果需要维护标签，请前往 标签列表 。

应用详情

在应用列表页点击应用 ID，即可进入该应用的详情页面，包括如下模块：

- **性能概览**：性能概览模块展示该应用的关键性能指标，包括 LLM 调用次数、LLM 调用耗时、模型调用次数、模型调用耗时、模型调用错误率、Token 使用量等。
- **Token 分析**：基于 Input/Output、模型调用、LLM 调用等维度，深度分析该应用的 Token 使用情况。
- **链路追踪**：链路追踪用于实现跨应用的多维度调用链检索与分析，您可以根据多种过滤条件组合进行调用查询，查询结果中的每一条记录代表一次调用，等同于一个 Span。您可以单击对应的 TraceID 进入链路详情视图，进一步分析链路中的每个环节。

性能概览

最近更新时间：2025-11-28 11:09:22

性能概念用于分析该业务系统内所有 LLM 应用的整体性能表现。

操作步骤

1. 登录 腾讯云可观测平台 。
2. 在左侧菜单栏中选择 LLM 可观测 > 性能概览 。
3. 通过右上方的时间选择器指定查询时间跨度。

指标说明

指标名称	说明
LLM 调用次数	统计指定时间范围内，应用发起的 LLM 服务调用总次数，直观反映 LLM 应用的业务访问热度。
LLM 调用平均耗时	指定时间内所有 LLM 调用的总耗时除以调用次数，是衡量 LLM 服务整体响应效率的基础指标。
LLM 调用耗时 P99	将指定时间内的 LLM 调用耗时按升序排列后，第99百分位对应的耗时值，反映99%的调用不会超过的耗时上限，用于评估极端场景下的性能。
LLM 调用耗时 P95	按升序排列所有 LLM 调用耗时后，第95百分位对应的耗时值，代表95%的调用耗时不超过该数值，体现大部分场景下的性能稳定性。
LLM 调用耗时 P50	又称中位数耗时，按升序排列所有 LLM 调用耗时后，第50百分位对应的耗时值，反映 LLM 调用的平均性能基准水平。
模型调用次数	统计指定时间内，LLM 模型的调用总次数，体现模型的实际使用频率。
首 Token 平均耗时	从发起 LLM 调用到接收返回的第一个 Token 的平均时间，是衡量 LLM 响应即时性的核心指标，直接影响用户交互体验。
模型调用平均耗时	指定时间内大模型调用总耗时除以模型的调用次数，用于衡量模型的整体运行效率。
模型调用错误率	指定时间内模型调用失败的次数占总调用次数的比例，直观反映 LLM 模型服务的稳定性与可用性。
Token 使用	统计指定时间内 LLM 调用过程中输入 Token 与输出 Token 的总数量（或分别统计），是计算模型使用成本与评估资源消耗的关键指标。

链路追踪

最近更新时间：2025-11-28 11:09:22

链路追踪用于实现跨应用的多维度调用链检索与分析，可以根据多种过滤条件组合进行调用查询，查询结果中的每一条记录代表一次调用的记录，等同于一个 Span。您可以单击对应的 TraceID 进入链路详情视图，进一步分析链路中的每个环节。

操作步骤

1. 登录 腾讯云可观测平台 。
2. 在左侧菜单栏中选择 LLM 可观测 > 链路追踪 。
3. 选择合适的地域以及业务系统。
4. 通过右上方的时间选择器指定查询时间跨度，通过查询对话框指定更多查询条件。

字段名	说明
应用名称	应用名称通常和服务名保持一致。应用名称在应用接入时指定，多个使用相同应用名称接入的进程，在 LLM 可观测中会表现为相同应用下的多个实例。
操作类型	操作类型是标记 LLM 应用具体操作环节的字段，主要用来区分 LLM 交互过程中的不同业务动作与技术步骤，包括 Agent、Chain、Task、Tool 等类型。
响应时间	可输入具体时间范围或阈值，筛选出符合该响应时间条件的 LLM 链路，快速定位性能瓶颈链路。
接口	接口名称等同于 Span 名称。
状态	表示该调用是否正确。

链路详情

在链路追踪的查询结果中，第一列展示了 TraceID，单击 TraceID 对应的链接，即可进入**链路详情**页面，通过瀑布视图洞察该链路中每一个环节的调用耗时以及执行状态。请参考 链路详情 了解更详细的操作说明。

模型分析

最近更新时间：2025-11-28 11:09:22

LLM 可观测的模型分析能力，聚焦全量模型的运行状态与使用效能，实时统计模型调用次数、平均耗时、错误率及 Token 使用量等核心指标，精准定位模型性能瓶颈与资源浪费问题，为模型优化迭代、资源配置调整提供数据支撑。

操作步骤

1. 登录 腾讯云可观测平台 。
2. 在左侧菜单栏中选择 LLM 可观测 > 模型分析 。
3. 在页面顶部选择合适的地域、业务系统以及模型。
4. 通过右上方的时间选择器指定查询时间跨度。

指标说明

指标名称	说明
模型调用次数	统计指定时间内，LLM 模型的调用总次数，体现模型的实际使用频率。
首 Token 平均耗时	从发起 LLM 调用到接收返回的第一个 Token 的平均时间，是衡量 LLM 响应即时性的核心指标，直接影响用户交互体验。
模型调用平均耗时	指定时间内大模型调用总耗时除以模型的调用次数，聚焦模型的整体运行效率。
模型调用错误率	指定时间内模型调用失败的次数占总调用次数的比例，直观反映 LLM 模型服务的稳定性与可用性。
Token 使用	统计指定时间内 LLM 调用过程中输入 Token 与输出 Token 的总数量（或分别统计），是计算模型使用成本与评估资源消耗的关键指标。

资源管理

最近更新时间：2025-12-05 15:19:52

资源管理是 LLM 可观测和应用性能监控（APM）的通用能力，用于管理业务系统、查询用量，以及管理预付费套餐包。

操作步骤

1. 登录 腾讯云可观测平台 。
2. 在左侧菜单栏中选择 LLM 可观测 > 资源管理，选择需要新建业务系统的地域。

模块说明

业务系统管理

LLM 可观测复用应用性能监控（APM）的业务系统，实现应用分类管理。每个业务系统有唯一的 Token，应用接入的时候需要指定 Token。可以在业务系统级别设置存储时长、计费方式等参数，也可以基于业务系统实现权限管理和分账。不同业务系统之间的监控数据完全隔离。关于业务系统的划分，请参见 如何划分业务系统 。
请参考如下配置对业务系统进行维护。

配置项	说明
业务系统名称	自定义业务系统名称。
开启免费模式	开启免费模式后，该业务系统将永久免费。关于免费模式的限制，请参见 关于免费模式的使用限制 。
计费模式	支持**按量付费**和**预付费**。
上报地域	各地域数据隔离，业务系统创建后不可更改。
链路存储时长	支持选择1天、3天、7天、15天、30天链路数据存储时长，试用期间默认存储时长为1天。存储时长越长，收费越高。
业务系统简介	可以简单描述业务系统用途等。
添加标签	可以结合腾讯云资源标签，对业务系统打标，以实现按标签授予权限和按标签分账功能。请参见 访问管理 设置标签。

关于业务系统的更多详情，请参见 新建业务系统 。

用量分析

LLM 可观测复用应用性能监控（APM）的计费方式，通过Span 上报数（条）、Span 存储量（条*天）、探针在线时长（探针个数*小时）三个维度体现使用量。通过用量分析页面，您可以分析每个业务系统产生的使用量，以更好地评估 LLM 可观测费用。
关于用量分析的更多详情，请参见 用量分析 。

套餐包管理

LLM 可观测复用应用性能监控（APM）的套餐包机制，在特定场景下，使用套餐包能降低产品费用，您可自行选择购买。套餐包可同时抵扣应用性能监控（APM）和 LLM 可观测的用量。在套餐包管理页面，可以查看您已购买的套餐包，以及套餐包的使用情况。
关于套餐包的更多详情，请参见 套餐包介绍 。