

中国DevOps社区峰会 2024 · 上海

10.19 | 上海龙之梦大酒店



源于社区 服务社区

 中国DevOps社区峰会 2024 · 上海



解决企业个性化代码生成准确率的核心实践

徐磊 英捷创软 首席架构师





徐磊

- AISE 首架构师/产品经理
- 微软最有价值专家MVP，微软区域技术总监Regional Director
- GitHub Star / GitHub Copilot 中国区授权服务资深顾问
- 开放原子开源项目SmartIDE 创始人/核心贡献者
- 华为云MVP
- 资深软件工程/敏捷/精益/DevOps专家，EXIN认证DevOps Master/Professional/认证讲师
- 书籍作者和译者《专业SCRUM 基于Azure DevOps的敏捷实践》，《云原生应用开发实践》，《基础设施即代码 - 模式与实践》



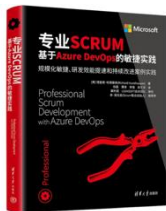
公众号：数字共生
与数字世界共生进化



公众号：DevOps
企业数字化转型实践



Microsoft
Regional Director



中国DevOps社区峰会 2024 · 上海

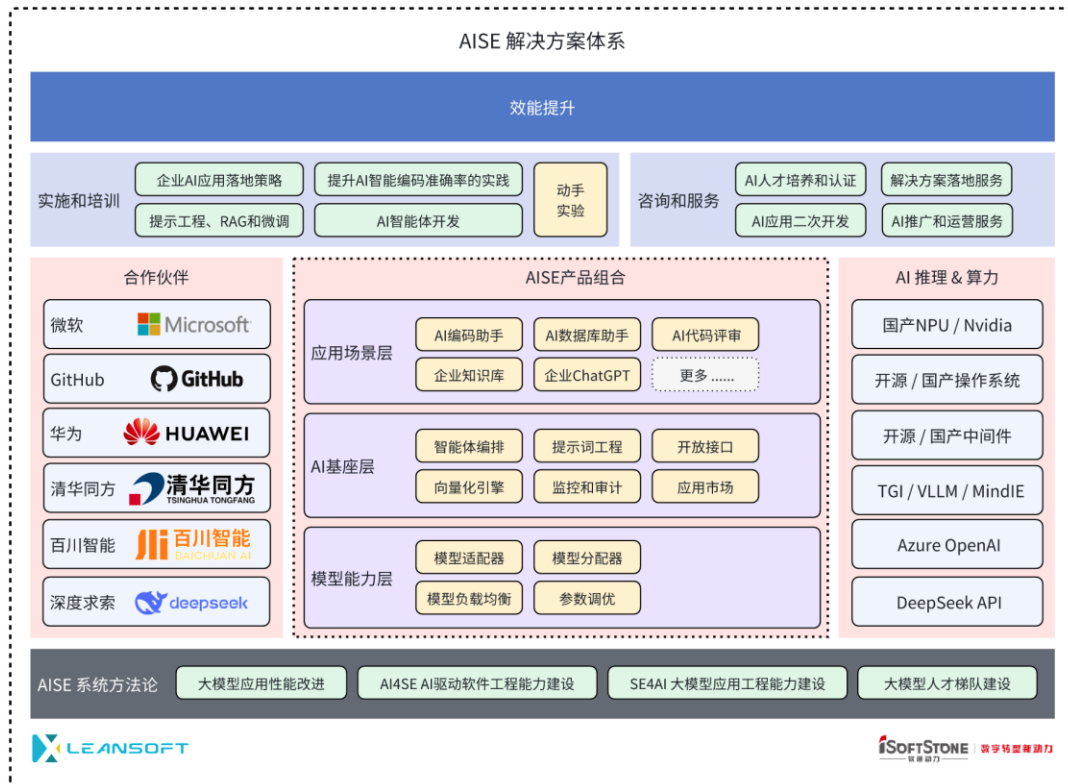
DevOpsChina.org



源于社区 服务社区



AISE - 企业落地AI大模型的端到端解决方案



- AISE方法论提供体系化理论支撑
- AISE产品组合提供最佳实践落地的平台和工具
- 微软 Azure OpenAI 完整支持
- GitHub Copilot 扩展支持
- AISE已经完整适配 国产NPU、通用硬件、操作系统和中间件
- 华为昇腾 910/310 提供端到端性能和优化
- 推理服务全栈支持：TGI、VLLM、MindIE（华为）
- DeepSeek API 完整支持
- 体系化实施、培训和咨询服务，为企业落地AI提供端到端保驾护航



案例：汇丰科技

- 2023.6 项目启动
- 2023.8.9 完成 AISE v0.3版本 内部部署
- 2023.8.20 全球CIO访华期间完成产品演示，受到认可，准许推广
- 2023.9 完成 AISE v0.4版本 内部部署
- 2023.10 月底推广到 个人银行事业部中国区3500人使用，
- 2023.10 完成模型访问安全性提升和多实例负载均衡改造
- 2023.11 效能仪表盘上线、提供代码生成和接受率指标
- 2023.12 提示词库v2(多模型混合调度) 和RAG能力上线

我们监控了什么？

◆ Code contribution (Line-of-code)

◆ Code commits

◆ Pull Requests (PR)

◆ PR review lead time

平均每10人每两周

32.7%

Code contribution (LoC)



20.7%

Pull requests (PR)



95.7%

Code commits



- 10%

PR review lead time



案例：博时基金

- 定制版插件
- 最早适配910B算力
- K8s集群部署
- 协助推广，培训
- 内部运营



iSOFTSTONE 智慧软件助力

软动力 | AISE & SmartCode

AI智能编码

提高代码生成准确率 实战体验

微软最有价值专家 & GitHub Star 承接



课程内容

- ☑ 代码补全
- ☑ 提示词构造
- ☑ 代码对话
- ☑ 提示工程技巧
- ☑ 自定义提示词
- ☑ 真实项目实战演示

本课程是 AISE 系统为完成的一部分 目标实践





编程助手 smartcode 全新升级

支持 JetBrains / VS Code

更有效 更灵活 更稳定

编程助手 AISE 主要功能包括代码智能补全、代码解释、代码生成、技术咨询等，能有效提升我们的编程效率和质量。

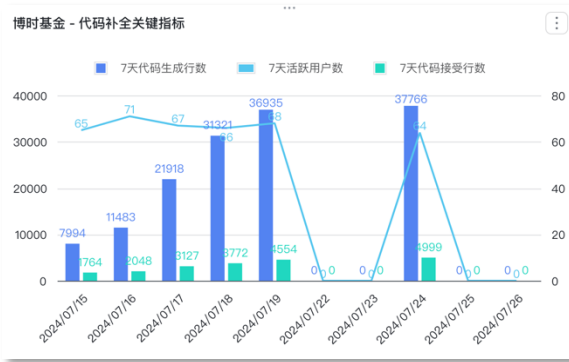
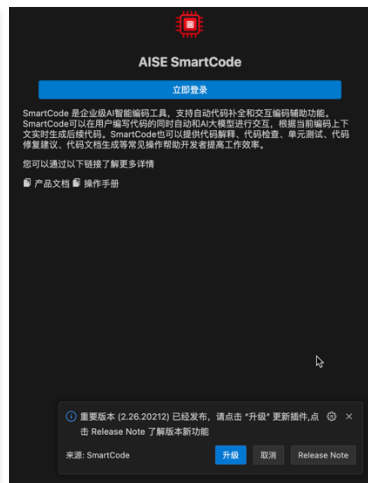
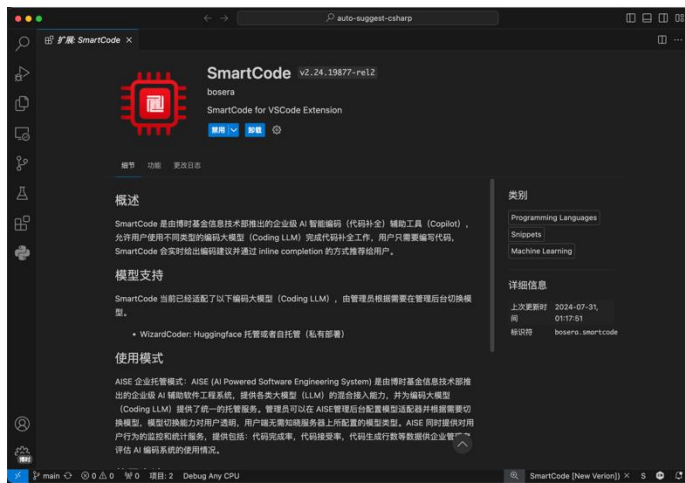
主要升级功能

- 效果更好**
 - 升级为更为强大的 Deepseek 模型，代码补全效果更好。
- 使用更灵活**
 - 支持更灵活切换单行和多行补全模式。
 - 支持取消自动补全。
 - 支持设置快捷键补全。
- 私有定制**
 - 支持私有定制，右键 code check 集成公司 Java 规范，代码检查效果更佳。

更多功能说明

- 支持昇腾显卡，算力充足。
- 两个插件合二为一，安装更便捷，兼容性更好。
- 提供清晰详尽的产品文档，帮助用户快速上手和解决问题。

请注意：请先卸载原有的2个插件（smartchat, smartcode）后再安装！



目录

- 1 代码生成中的提示工程原理
- 2 企业真的需要私有化代码训练吗?
- 3 面向开发人员的提示工程技巧
- 4 从智能副驾到智能体 – 代码生成技术路线
- 5 Q&A



解构一段真实的 代码补全 的提示词

解构目标：通过了解 构建提示词的过程，优化我们使用 SmartCode 的方式，以便获得最佳的代码生成效果。

PROMPT - PREFIX

```
// Path:
src/main/java/org/example/corebanking/BankAccount.java\r\n
// Compare this snippet from
src/main/java/org/example/corebanking/BankAccountController.java:
a:\r\n\r\n// package org.example.corebanking;\r\n\r\n// import
org.springframework.web.bind.annotation.RestController;\r\n\r\n//
import java.util.HashMap;\r\n\r\n// import java.util.Map;\r\n\r\n//
\r\n\r\n// @RestController\r\n\r\n// public class BankAccountController
{\r\n\r\n// \r\n\r\n// }\r\n\r\n\r\n\r\n
package org.example.corebanking;\r\n\r\npublic class BankAccount
{\r\n private String accountNumber;\r\n private String
accountName;\r\n private double balance;\r\n
```

PROMPT - SUFFIX

FIM提示词格式

- 前文
- 后文

关键元素：

- 当前文件名和完整相对路径
- 相关文件名和完整相对路径、
相关文件中的代码片段
- 当前文件内容
 - 光标前的内容
 - 光标后的内容

什么是FIM?

Input: <fim_prefix>白日依山尽<fim_suffix>欲穷千里目<fim_middle>

Output: ???



Enable Physics: ☒

Show selected module types:

☒ Prompt generation ☒ Model post-processing ☒ Telemetry ☒ Auth ☒ Network ☒ Utility

☒ Library ☒ Unclassified

The graph displays a dense network of nodes and edges. The nodes are colored according to their module type: green for Prompt generation, blue for Model post-processing, orange for Telemetry, yellow for Auth, dark blue for Network, and light orange for Utility. The graph is highly interconnected, with a large central cluster and several smaller clusters. A large blue rectangular area is visible in the bottom right corner, likely a redaction or a placeholder for a label.




```

    threshold: -1,
    numberOfSnippets: 0,
  },
  conservative: {
    matcherFactory: M_jaccard_scorer.FixedWindowSizeJaccardMatcher.FACTORY
(10),
    threshold: 0.3,
    numberOfSnippets: 1,
  },
  medium: {
    matcherFactory: M_jaccard_scorer.FixedWindowSizeJaccardMatcher.FACTORY
(20),
    threshold: 0.1,
    numberOfSnippets: 2,
  },
  eager: {
    matcherFactory: M_jaccard_scorer.FixedWindowSizeJaccardMatcher.FACTORY
(60),
    threshold: 0,
    numberOfSnippets: 4,
  },
  eagerButLittle: {
    matcherFactory: M_jaccard_scorer.FixedWindowSizeJaccardMatcher.FACTORY
(10),
    threshold: 0,
    numberOfSnippets: 1,
  },
};

exports.getNeighborSnippets = async function (
  curFile,
  relevantDocs,
  neighborTabsOpt,
  indentationMinLengthOpt,
  indentationMaxLengthOpt,
  snippetSelectionOpt,
  snippetSelectionK
) {
  const nbrOpt = exports.neighborOptionToSelection[neighborTabsOpt];

  // nbrMatchers is an instance of JaccardMatcher (set to compare to curFi
  const nbrMatcher = (function (

```

Module	neighbor-snippet-selector(3055125)
Module Name:	neighbor-snippet-selector
Module Category:	<div> <div></div> <div>Prompt generat...</div> </div>
Exports	
<ul style="list-style-type: none"> getNeighborSnippets neighborOptionToSelection 	
Imports	
<ul style="list-style-type: none">  language-marker-constants  jaccard-scorer 	
Imported by	
<ul style="list-style-type: none">  get-prompt-actual 	

GitHub Copilot 源码浏览器（反编译版本 - 仅供学习研究）
<https://thakkarparth007.github.io/copilot-explorer/posts/copilot-internals.html>

代码补全提示词工程 全链路

IDE环境

```
leaniss-system-core

J AiseOssFileDetailServiceImpl.java 1, M • J AiseSessionDetailServiceImpl.java 2
impl > J AiseOssFileDetailServiceImpl.java > AiseOssFileDetailServiceImpl > avatar(MultipartFile, String, Long)

107 *
108 * @param fileId 文件信息明细主键
109 * @return 结果
110 */
111 @Override
112 public int deleteAiseOssFileDetailByFileId(Long fileId)
113 {
114     return aiseOssFileDetailMapper.deleteAiseOssFileDetailByFileId(fileId);
115 }
116
117
118 @Override
119 public AjaxResult avatar(MultipartFile file, String sessionId, Long userId) {
120     R<SysFile> fileResult = remoteFileService.upload(file);
121     if (StringUtils.isNull(fileResult) || StringUtils.isNull(fileResult.getData()))
122     {
123         return AjaxResult.error(msg:"If the file service is abnormal, contact the
124             administrator to check the file service status");
125     }
126     String url = fileResult.getData().getUrl();
127     String contentType = fileResult.getData().getContentType();
128     if (StringUtils.isEmpty(contentType))
129     {
130         contentType = "application/octet-stream";
131     }
132     String fileName = url.replace(minioConfig.getUrl() + "/" + minioConfig.
133         getBucketName() + "/", replacement:"");
134     AiseOssFileDetail aiseOssFileDetail = new AiseOssFileDetail();
135     aiseOssFileDetail.setSessionId(sessionId);
136     aiseOssFileDetail.setUserId(userId);
137     aiseOssFileDetail.setMinioObjectKey(fileName);
138     aiseOssFileDetail.setFileName(file.getOriginalFilename());
139     aiseOssFileDetail.setContentType(contentType);
140     aiseOssFileDetail.setFileSize(file.getSize());
141
142     if (insertAiseOssFileDetail(aiseOssFileDetail)>0)
143     {
144         AjaxResult ajax = AjaxResult.success();
145         ajax.put(key:"fileId", aiseOssFileDetail.getFileId());
146         return ajax;
147     }
148 }
```

上下文捕捉阶段

IDE 环境属性

前文

上下文生成器

- Coding language
- Open files
- Referenced files
- Cursor location
- Scoping
- Cache

文字提示

后文

代码提示词整理

根据当前鼠标光标的代码上下文位置, 以及IDE里打开的其他代码文件作为辅助参数, 再通过特定的算法 (该算法决定了Prompt的内容, 以及代码补全的质量) 对Prompt的内容进行整理、裁剪

补全逻辑处理阶段

时延处理

为了避免频繁想后台发送请求, 使用了防抖函数进行请求的时延处理。避免频繁的向模型发送请求, 提升客户端体验

多级缓存

使用缓存技术, 避免了多余、重复请求, 提升了代码补全的效率。降低了tokens的成本

模型参数调优

根据不同模型进行模型参数调优, 提高模型的响应速度、以及代码补全的质量。

客户端体验处理

为了避免频繁的给用户提供无效的补全, 补全插件会记录之前没有采纳的结果, 同时计算在当前上下文代码补全内容被采纳的可能性, 然后决定补全内容是否展示给客户端用户

LLM请求阶段

SSE处理

根据特定的算法, 对大语言模型的输出结果进行过滤与截取, SSE流式处理优化

遥测处理

通过代码补全服务端为客户端提供用户登录、授权、行为分析、统计等能力 插件使用人数、代码补全行数、代码补全方法数、编码总时长, 补全 占比等数据进行统计



源于社区 服务社区



目录

- 1 代码生成中的提示工程原理
- 2 企业真的需要私有化代码训练吗?
- 3 面向开发人员的提示工程技巧
- 4 从智能副驾到智能体 – 代码生成技术路线
- 5 Q&A



//SE4AI// 企业进行模型训练的6个可行性前提

案例：微软O产品团队基于GPT 3.5 模型微调支持个性化代码生成

扫码阅读详细分析



- ❑ **团队规模巨大：** 即便只有几个百分点的准确度提升，放大到几千人规模的开发团队上，产生的效能提升和节省的成本/资源也是非常巨大的。
- ❑ **单一代码库：** 几千人规模的O产品开发团队全部都在一种开发语言和非常类似的代码库上进行工作，因为任何的模型只要进行了再次训练，就只能针对某一种开发语言提供服务，再次训练虽然会增强模型在O产品代码库上的能力，同时也会降低这个模型实例对其他编程语言的处理性能。这一点上，O产品团队符合要求。
- ❑ **代码足够特殊：** 特殊到GPT 3.5 Tubro 模型在训练的时候完全没有见过类似的代码。基于前面分析的，虽然是C/C++语言，但是因为语言本身的高度灵活性和这个单一产品本身的规模，其代码库中代码是GPT完全没有见过的。
- ❑ **代码规模巨大：** 要通过训练对基座模型产生影响，至少需要几百万行规模的高质量单一语言代码。面对一个几十/上百亿参数量的模型，如果训练数据不够，则不足以影响模型的行为。即便采用非常高效的微调方式，也需要原有数据的1%左右规模的数据才能发挥作用。
- ❑ **容忍训练失败：** 即使在类似GPT这样高质量的模型上进行再次训练并且由非常有经验的工程师来操作，如果参数控制不到位也容易出现失败的情况。加上每次训练所需要的算力和时间消耗，这是一个非常高风险的投入。
- ❑ **足够的不可变基础代码：** 模型训练会让模型产生非常强的偏好，要修改这个偏好就必须重新训练，因此用来训练的代码必须是那些在很长时间内（至少几年）中不会发生变化的代码。如果私有代码组件已经发生了改变，而模型还在一直生成老的代码，反而会对工程师的工作效率产生负面影响。O产品因为有大量自行研发的可重用的基础组件代码，也符合要求。



//SE4AI// 大模型应用工程能力建设 之 1-2-3

使用工程化方法加速大模型应用的迭代和交付频率, 提升性能, 质量和用户体验

1个目标

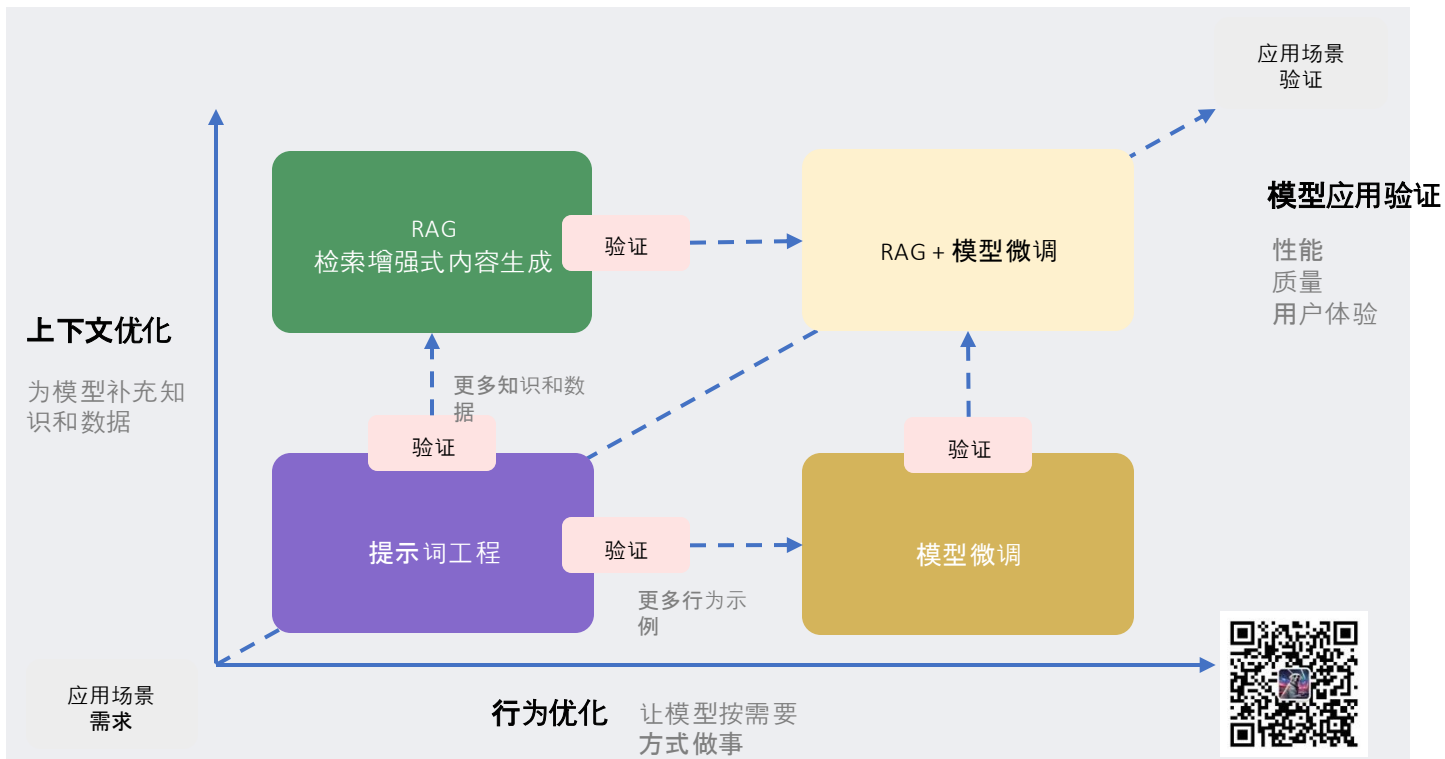
从应用场景出发，使用客观可度量的验证标准，确保应用场景的性能、质量和用户体验。

2个维度

- 上下文优化：为模型补充数据和知识
- 行为优化：为模型提供更多行为示例

3个方法

- 提示词工程：经济、高效的应用优化路径；快速验证场景可行性，提供初期应用原型验证
- RAG：提高模型对私域数据的额认知能力。
- 模型微调：提高模型响应的规范性和标准化，提供可靠的应用输出性能。



//SE4AI// 提示工程、RAG和微调 – 如何选择



扫码阅读详细分析

- 提示工程可以同时解决知识和行为的问题，但是会受限模型数据窗口大小并引发响应缓慢
- RAG能解决知识补充问题但是不擅长改变模型行为
- 微调可以改变模型行为但是不适合为模型补充知识

上下文优化

这个纬度主要关注私域数据，是让模型知道它所不知道的事情，包括：模型训练中从未见过的数据，比如内部代码、文档、规范、策略等。**这个维度主要解决模型输出内容上的相关性。**

起点

提示工程

验证

RAG
检索增强式内容生成

验证

模型微调

验证

RAG + 模型微调

上下文+行为优化

可以同时为模型补充上下文（上下文优化）和优化模型的行为（行为优化）。因此提示词工程可以同时在这2个维度上帮助我们提升模型的性能、质量和用户体验，让我们可以更快达到目标。

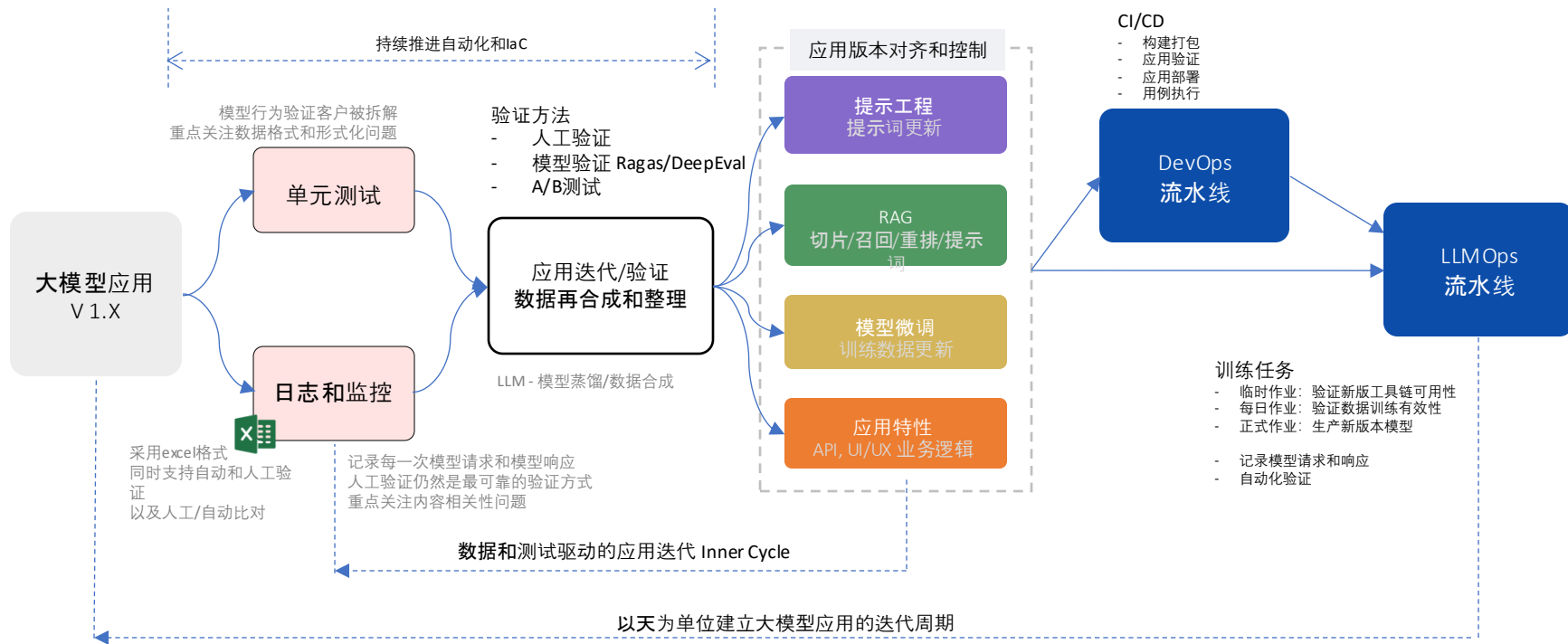
行为优化

这个维度主要关注模型的行为，是教会模型去按照我们希望方式做事，包括：输出内容的格式、语气、偏好；甚至生成固定格式的请求以便调用其他服务。**这个维度主要解决模型输出形式上的稳定性。**



//SE4AI// 大模型应用的持续交付

建立系统化的大模型应用改进路径，用工程化方法提升内容相关性和输出稳定性





数字化是智能化的基础设施
没有完善的DevOps工具和实践体系
AI赋能软件开发提效只能是空中楼阁



目录

- 1 代码生成中的提示工程原理
- 2 企业真的需要私有化代码训练吗?
- 3 面向开发人员的提示工程技巧
- 4 从智能副驾到智能体 – 代码生成技术路线
- 5 Q&A



提示工程核心原则 – 4S

提示工程的目标是：**制作清晰的指令**，以指导 AI 系统根据项目的**特定需求生成符合上下文的代码**。确保代码在**语法、功能和上下文 方面都正确**。

高效提示词 = 单一具体的 **意图** + 足够的 **上下文**

越短越好（不要过渡词、不要连接词、不需要客套、不需要解释），直接表达意图

遵循提示工程的**4S原则**构建自己的提示词

Single（单一）	始终将提示重点放在单个明确的任务或问题上。这种清晰度对于获得准确有用的答复至关重要。
Specific（具体）	确保你的指令明确且详细。越是具体的提示词越会获得更适用和精确的代码建议。
Short（简短）	在具体的同时，请保持提示简明扼要。这种平衡既能确保清晰度，又不会使 AI 负担过重或使交互复杂化。
Surround（有上下文）	利用描述性文件名并使相关文件保持打开状态。这为 AI 提供了丰富的上下文，从而产生更有针对性的代码建议。

*用户需要了解和遵循**4S原则**协助构建准确的提示词*





4S原则 – 反模式和模式示例

明确单一目标 (Write a python function),
简短的描述

反模式	模式
#Give me an even number list	#Write a Python function to filter and return even numbers from a given list
# create an Express app, that uses TypeScript and Pug, and that has a products page that retrieves data from a MongoDB database.	#create the Express app with TypeScript and Pug #add a products page # retrieve the customer data from a database





4S原则 – 反模式和模式示例

给出更加丰富的上下文（**Surround**），可以让**SmartCode**完成更加复杂的任务
以下右侧示例中：我们给出了更加明确的 **step by step** 指令
确保**SmartCode**按照我们的要求完成生成任务

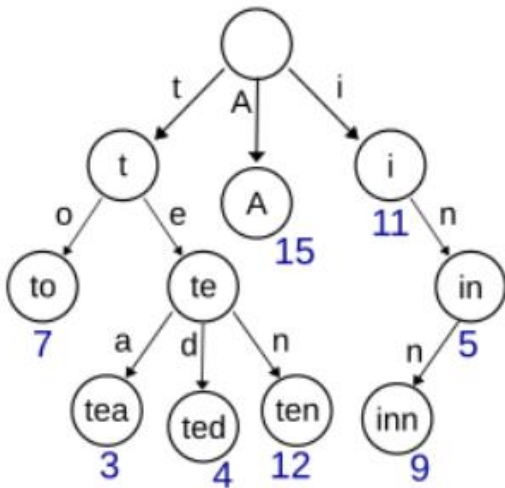
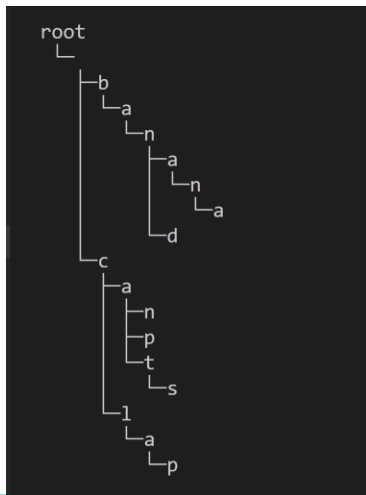
反模式	模式
<pre># write a simple flask app that returns a list even number from a list of number, you need to create a function that take a list of numbers and this function should return the even values. Please start by create a sample list with numbers, then create a list of even numbers from the sample list. Finally, you need to return the list of even numbers</pre>	<pre># write a simple flask app that return a list of even numbers from a list of numbers # create a function that take a list of numbers and return the even values # create a sample list of numbers # create a list of even numbers from the sample list # return the list of even numbers</pre>




动手实验：使用 SmartCode 开发自动推荐系统

作为一名资深的开发人员，你刚刚接手了这个叫做 **自动推荐系统 (auto-suggestion)** 的项目。这个项目的特性是**通过存储一个大量单词的词库，实现快速搜索与给定前缀匹配的字词，以提供自动完成功能。**另外，这个项目还支持对拼写错误的单词实施拼写建议。

你已经拿到了一个使用 **字典树 (trie)** 数据结构的现有代码库，该结构是一种基于树的数据结构，对于前缀匹配和自动完成非常有用。你需要快速熟悉这个代码库并对其进行扩展才能满足项目组对于自动推荐系统的需求。



软通动力 | 软通动力

软通动力 | AISE & SmartCode

AI智能编码

提高代码生成准确率

实战体验

微软最有价值专家 & GitHub Star 亲授



课程内容

- ☑ 代码补全
- ☑ 代码对话
- ☑ 自定义提示词
- ☑ 提示词解析
- ☑ 提示工程技巧
- ☑ 真实项目实战演示

本课程是 AISE 系统方法论的一部分 扫码获取





动手实验：使用 SmartCode 开发销售报表系统

处理数据是大多数商业应用中最重要的一部分，数据往往代表某种特定的业务对象，比如：用户，产品，订单，销售额。商业应用中的业务逻辑也和数据直接相关。SmartCode 理解和处理数据的能力直接关系到 生成式AI 在业务开发场景中的实战价值。

在本示例中，我们将以 销售报表系统 为例，利用 SmartCode 完成数据结构建模，样例数据生成，数据统计和数据展示这几个环节的代码生成。通过这个示例，开发者可以了解使用 SmartCode 处理业务数据和业务逻辑的关键实践和技巧。我们将在不同的关节中使用 代码补全 和 Chat 能力来完成不同的代码生成任务，

最终构建一份
清晰可读的销售报表

Quarterly Sales Report

Q1: Sales: \$2,147,913.85, Profit: \$259,780.34, Profit Percentage: 6.00%
By Department:

Department	Sales	Profit	Profit Percentage
Accessories	\$250,965.22	\$32,137.16	10.00
Children's Clothing	\$240,996.59	\$30,624.26	20.00
Footwear	\$350,495.72	\$44,661.94	17.00
Men's Clothing	\$179,416.69	\$19,491.29	13.00
Outerwear	\$341,525.29	\$47,556.50	16.00
Sportswear	\$216,981.82	\$21,858.47	7.00
Undergarments	\$298,357.33	\$35,451.12	6.00
Women's Clothing	\$269,255.19	\$27,999.61	20.00

Top 3 Sales Orders:

Product ID	Quantity Sold	Unit Price	Total Sales	Profit
SPRT-702-S-BK-CA1	96	\$295.33	\$28,351.45	\$4,536.23
OUTR-609-M-GR-3P1	92	\$243.62	\$22,412.59	\$4,482.52
ACCS-437-S-GY-3P1	96	\$268.76	\$25,800.68	\$4,128.11

SOFTSTONE

软通动力

数字转型新动力

软通动力 | AISE & SmartCode

AI智能编码

提高代码生成准确率

实战体验

微软最有价值专家 & GitHub Star 亲授

课程内容

☒ 代码补全

☒ 代码对话

☒ 自定义提示词

☒ 提示词解构

☒ 提示工程技术

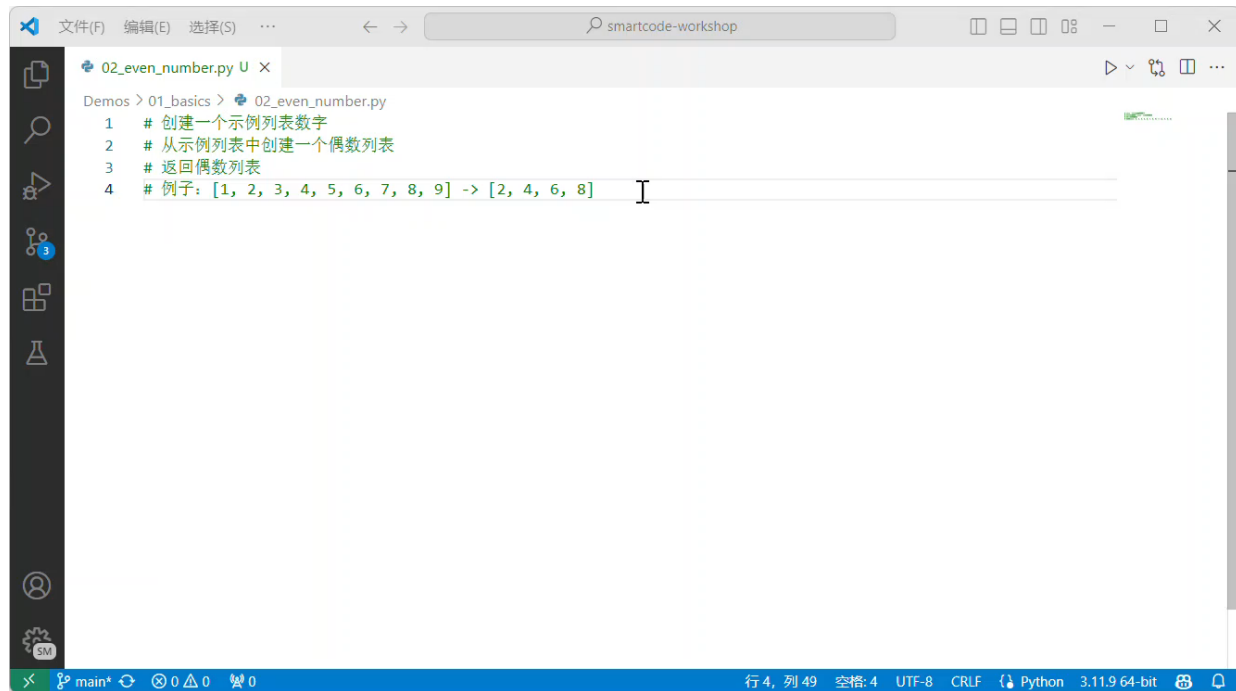
☒ 真实项目实战演示

本课程是
AISE系统方法论的一部分
扫码获取



示例 很重要

使用示例，包括代码风格，注释语言，数据结构等
给出示例可以非常直接的引导 SmartCode 生成我们需要的内容



```
02_even_number.py X
Demos > 01_basics > 02_even_number.py
1 # 创建一个示例列表数字
2 # 从示例列表中创建一个偶数列表
3 # 返回偶数列表
4 # 例子: [1, 2, 3, 4, 5, 6, 7, 8, 9] -> [2, 4, 6, 8]
```

利用示例为 **SmartCode** 提供更准确的上下文。

示例数据非常有效

比如：直接给出一个数字列表
比用语言描述这个列表可以更加准确的引导AI识别它需要处理的数据。
当这个数据非常复杂的时候，更加有用。



示例 很重要 – 零样本和 多样本学习

Zero-shot vs. Few-shot

通过类似代码片段引导SmartCode按我们的 编码风格 生成代码

通过场景化示例将 业务规则 和 边界条件 进行完整描述

```
1 # I need a function to convert Celsius to Fahrenheit
2
3 def celsius_to_fahrenheit(celsius):
4
5     fahrenheit = (celsius * 9/5) + 32
6     return fahrenheit
7
```

```
# Example: Function to convert Fahrenheit to Celsius
def fahrenheit_to_celsius(fahrenheit):
    celsius = (fahrenheit - 32) * 5/9
    return celsius

# Now I need a function to convert Celsius to Fahrenheit

def celsius_to_fahrenheit(celsius):
    fahrenheit = celsius * 9/5 + 32
    return fahrenheit
```

```
greeting.py X
greeting.py
1 # Example 1: Greeting message for morning
2 # Input: 9 AM
3 # Output: "Good morning!"
4
5 # Example 2: Greeting message for afternoon
6 # Input: 2 PM
7 # Output: "Good afternoon!"
8
9 # Example 3: Greeting message for evening
10 # Input: 7 PM
11 # Output: "Good evening!"
12
13 # Now, generate a python code that takes the current time as input using the datetime module
14 # and returns the appropriate greeting message
15
16 # Solution:
17 # Import datetime module
18 import datetime
19
20 # Get current time
21 current_time = datetime.datetime.now()
22
23 # Get current hour
24 current_hour = current_time.hour
25
26 # Check if it is morning (before 12 PM)
27 if current_hour < 12:
28     print("Good morning!")
29
30 # Check if it is afternoon (between 12 PM and 4 PM)
31 elif current_hour < 16:
32     print("Good afternoon!")
33
34 # Check if it is evening (after 4 PM)
35 elif current_hour < 21:
36     print("Good evening!")
37
38 # Else it is night time
39 else:
40     print("Good night!")
41
```

利用示例为
SmartCode 提供更准确的上下文。

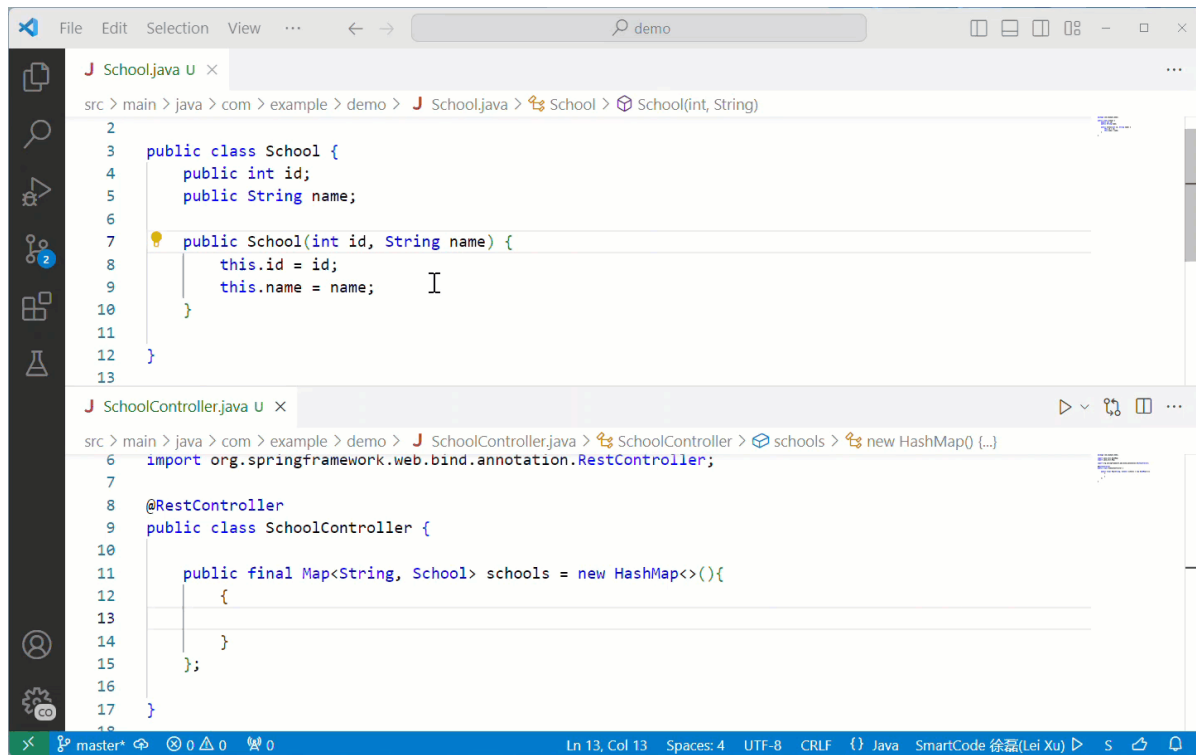
利用“多个”示例完整展示不同的输入/输出场景

给与AI足够的逻辑路径上下文。



RAG 很重要 – 相关代码文件的引入和识别可以大幅提升准确率

SmartCode 可以自动收集最多20个相邻文件的代码上下文作为当前代码生成的参考



```
src > main > java > com > example > demo > J School.java > School > School(int, String)
2
3 public class School {
4     public int id;
5     public String name;
6
7     public School(int id, String name) {
8         this.id = id;
9         this.name = name;
10    }
11
12 }
13

J SchoolController.java U X
src > main > java > com > example > demo > J SchoolController.java > SchoolController > schools > new HashMap() {...}
6 import org.springframework.web.bind.annotation.RestController;
7
8 @RestController
9 public class SchoolController {
10
11     public final Map<String, School> schools = new HashMap<>(){
12     {
13
14     }
15 };
16
17 }
```

利用RAG为SmartCode提供更准确的上下文。

代码是最好的示例

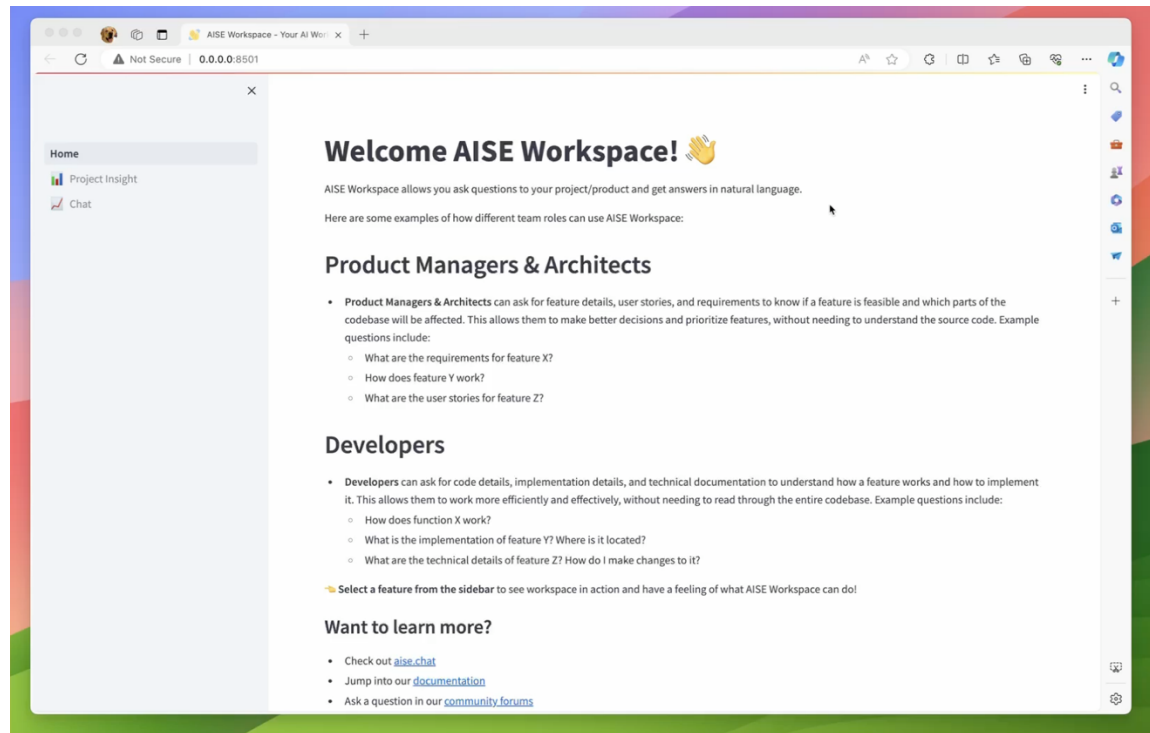
将与当前文件相关的代码打开，A自动参考这些代码完成当前文件的代码生成。

利用AST作为RAG检索逻辑，引入相关文件



RAG 很重要 – 构建代码和文本的双向量索引

预构建智能体利用RAG，向量数据库和为代码库，数据库，需求列表等特殊结构优化过的索引引擎



- 项目级/产品级问答
- 特性级代码分析和解决方案生成
- 面向产品，项目经理，架构师等非编码人员的工作入口
- 可以回答
 - 这个项目是做什么的？
 - 使用了哪些技术和框架？
 - 这个项目包含哪些api接口？
 - 如何实现XXX功能？
- 协助技术管理者获取技术和业务背景，加速解决方案设计。

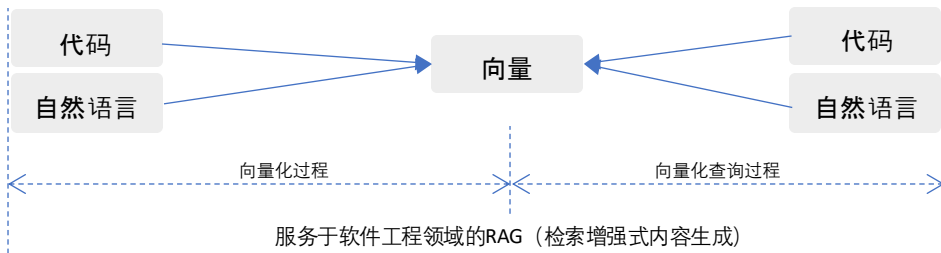


RAG 很重要 – 双向向量索引和传统IDE技术加持

- AISE内置代码/自然语言双向向量化模型、并提供内置的代码库索引引擎。
- AISE可以自动对指定代码库进行扫描，识别代码/文档/配置文件等不同文件类型，并建立多级代码库向量索引
- 同时引入基于tree-sitter的AST（抽象语法树）解析器，建立文件级/class级/方法级/代码块级/代码行级多级代码语义索引

步骤1 – 代码索引

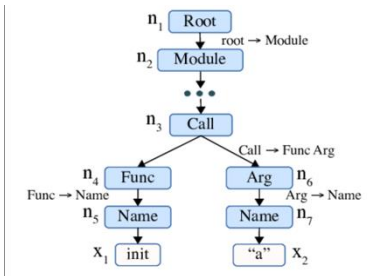
- 自然语言和代码
- 映射到同样的向量空间



- 提供自然语言和编程语言的双向向量相关性检索；
- 利用代码解析引擎构建基于AST的多级树状代码索引
- 利用LSP建立代码引用关系图谱，构建Graph增强的代码语义检索能力

步骤2 – 代码AST构建 使用tree-sitter构建抽象语法树(AST)

<https://tree-sitter.github.io/tree-sitter/>



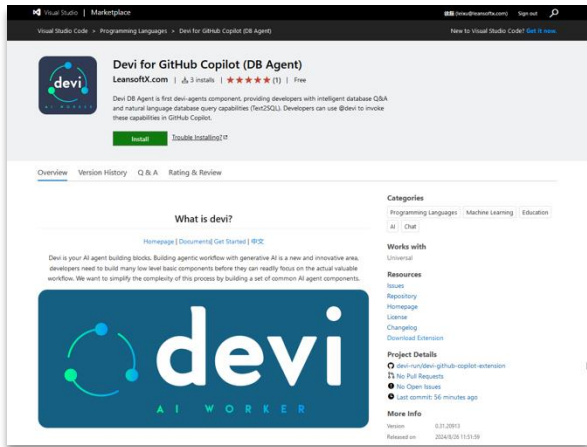
步骤3 – 分级检索

使用相似度算法定位代码到class/method/line级别

	sort_order	cosine_similarity	diff	type	start_row	end_row
3	0	0.701883	0.000000	method_declaration	30	36
2	1	0.682281	0.019601	method_declaration	18	20
5	2	0.672790	0.029093	method_declaration	57	63
4	3	0.656993	0.044890	method_declaration	45	47
0	4	0.634064	0.067819	program	0	97
1	5	0.632524	0.069358	class_declaration	9	96
6	6	0.617066	0.084817	method_declaration	71	73
7	7	0.494361	0.207522	method_declaration	82	84
8	8	0.457621	0.244262	method_declaration	93	95



数据库智能体 - Devi DB Agent for GitHub Copilot



Devi for GitHub Copilot 是构建在 GitHub Copilot 生态体系上的一系列 AI智能体，devi提供了这些智能体的原子能力，并通过 GitHub Copilot Chat 为开发者提供服务。

代维DB智能体 (devi DB agent) 是我们首批发布的智能体组件，可以为开发者提供智能数据库问答和 Text2SQL 的自然语言数据库查询能力。



扫码免费获取



通用对话：这个数据库是干啥用的？



通用对话：哪些表之间存在关联？



Text2SQL：获取所有review的汇总统计

aiise_app_version 表

列名	数据类型	约束	注释
app_version_id	bigint(20)	NOT NULL AUTO_INCREMENT	主键ID
app_id	bigint(20)	NOT NULL	应用主键
version_string	varchar(300)	DEFAULT NULL	版本描述
version_commit_id	varchar(30)	DEFAULT NULL	版本号
publisher	varchar(300)	DEFAULT NULL	发布者
release_date	datetime	DEFAULT NULL	发布日期
release_note	varchar(300)	DEFAULT NULL	版本说明
artifact_url	varchar(300)	DEFAULT NULL	附件网址
click_count	int(11)	DEFAULT '0'	下载次数
avatar	varchar(100)	DEFAULT ''	头像地址
create_by	varchar(64)	DEFAULT 'admin'	创建者
create_time	datetime	DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	创建时间
update_by	varchar(64)	DEFAULT NULL	更新者
update_time	datetime	DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	更新时间
remark	varchar(300)	DEFAULT NULL	备注
force_update	varchar(10)	NOT NULL DEFAULT 'true'	强制更新标志

注：表的主键是 app_version_id。

文档生成：给我XXX表的详细说明



中国DevOps社区峰会 2024 · 上海

DevOpsChina.org



源于社区 服务社区



AI辅助代码评审 – PR Agent

提供标准 WebHook API，方便DevOps平台对接

指令集

`/describe` 扫描PR，给出概述

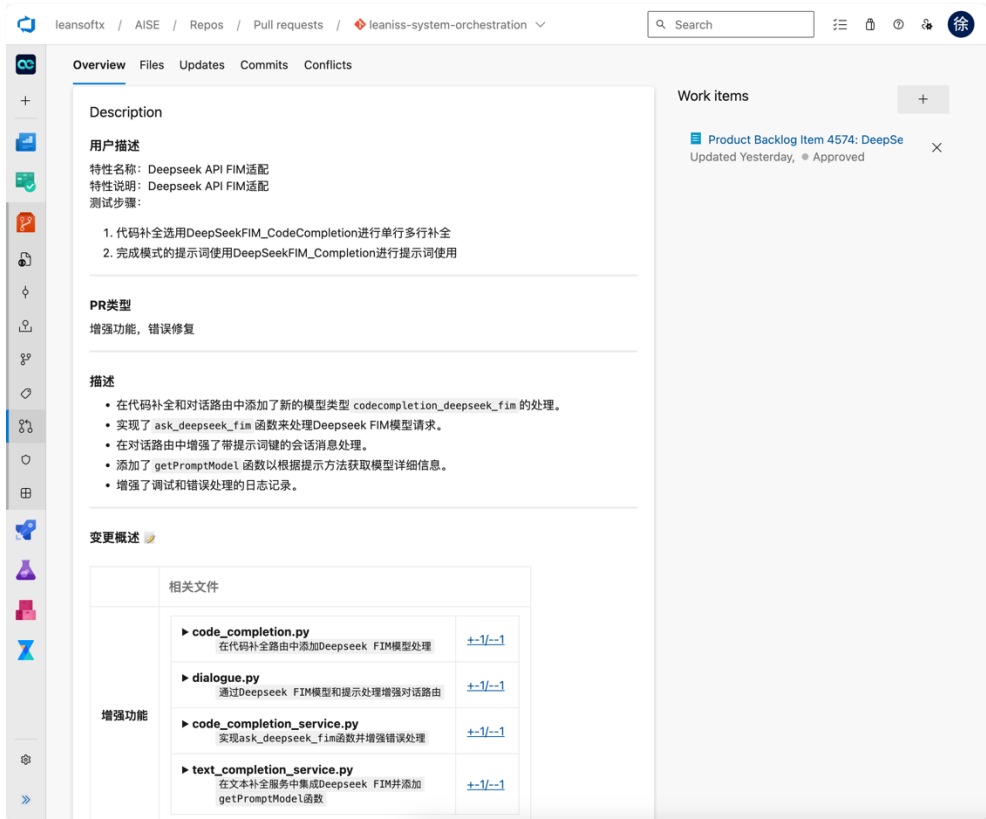
`/update_changelog` 生成标准的变更日志

`/review` 审核代码，给出修改建议

`/ask` 基于当前PR内容，解答问题

利用RAG技术，将PR代码变更集作为检索数据源，并根据需要引入相关的 需求/测试用例/任务 上下文；引导AI给出更加贴近企业个性化场景的审核建议。

作为AISE跳出编码环节的第一个探索场景，PR Agent 可以帮助代码评审者快速理解PR内容，以一个200行左右的代码变更为例，人工阅读并理解这些变更内容至少需要10分钟，利用AI进行总结之后只需要30秒即可理解。



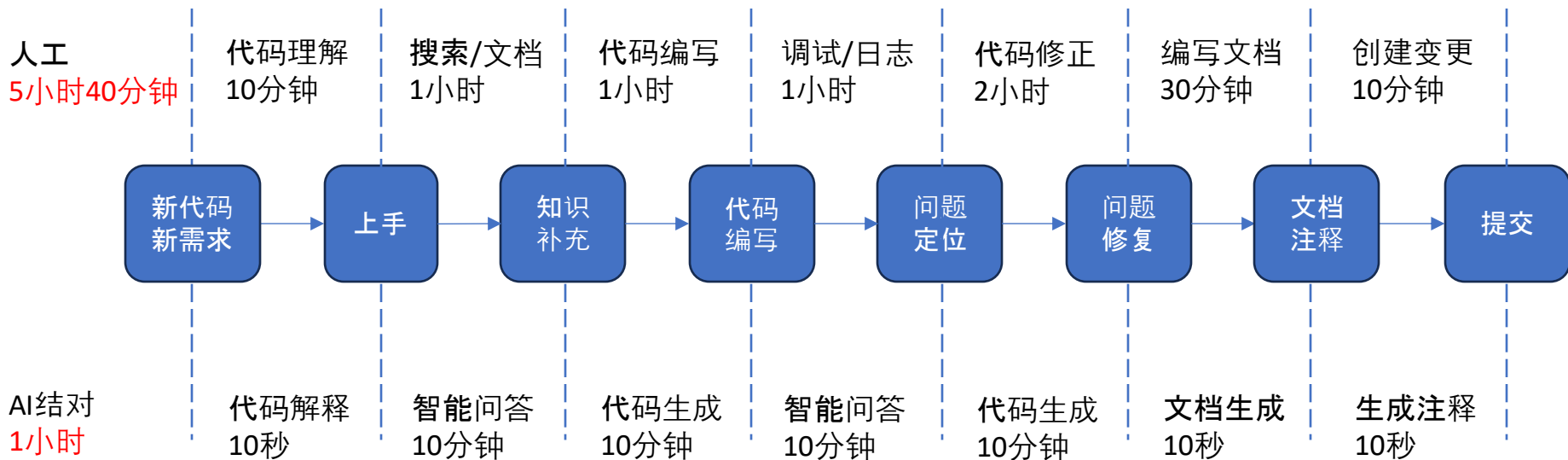


AISE的价值思考

- 为什么代码生成是GenAI的最早实现商业化成功的领域?
- 什么样的AI应用才不会被LLM吃掉?
- 如何识别那些引入AI即可迅速体现出价值的场景?



如何评估AI带来的研发效能



目录

- 1 代码生成中的提示工程原理
- 2 企业真的需要私有化代码训练吗?
- 3 面向开发人员的提示工程技巧
- 4 从智能副驾到智能体 – 代码生成技术路线
- 5 Q&A



AISE愿景：AI驱动的研发场景 端到端覆盖

- **需求分析** 需求自动拆分、任务自动拆分分配，自动生成验收标准；需求完整度、合规性验证
- **项目级认知** 智能对话
- **代码评审** 基于AI给出的代码评审意见，辅助完成源代码代码评审；打破技术鸿沟



产品/项目经理

- **接口测试** 自动生成
- **功能测试** 用例生成、步骤和预期结果自动生成
- **测试数据** 自动生成
- **缺陷定位** 缺陷分析和代码定位
- **代码检查** 检查结果解析和修复建议
- **代码安全** 漏洞分析和修复建议



测试/QA工程师

- ✓ 已发布
- 构建中
- 规划中

AI 辅助的端到端软件研发过程



市场人员

- **反馈分析** 用户行为分析，大量用户反馈数据处理
- **需求提取** 提取并生成高质量产品需求、改进点



设计师
(UI/UX)

- **原型迭代** 自动生成UI原型
- **原型到代码** 基于原型自动生成可用的界面代码(html/CSS) 组件



开发工程师

- ✓ **代码聊天** 辅助结对编程助理
- ✓ **代码补全** 代码补全和生成
- ✓ **单元测试** 自动生成
- **多模态辅助** 智能编码
- **数项目级认知** 智能对话
- **数据库认知** 智能对话
- **代码评审** 辅助团队写作和代码质量提升



运维工程师

- **部署和发布** IaC 辅助编写、生成和变更分析
- **日志分析** 关键信息提取
- **问题定位** 自动分析并定位到代码
- **智能客服** 根据内部知识库快速检索和相应用户问题，提供修复建议



AISE路线图：从智能副驾到智能体

企业数字化员工

智能体
AI native worker

AI copilot
智能副驾

AI 助理和代理
/ 记忆 / 知识图谱构建 / 数据连接器

AI 程序员和数字员工
/ 任务规划 / 代码生成和调试 / 目标识别和达成

AI 应用的持续交付 / 构建企业持续改进的数据飞轮

图表识别和生成 / 基于多模态的交互式开发辅助

项目级认知 / 基于RAG技术的交互式软件设计/分析/故障定位辅助

代码评审用例生成 / 交互式QA辅助

代码补全和智能对话 / 交互式编码辅助

人才

模型

算力

工作坊
理论+实践的实操性
体系化培训

灵活接入
SaaS服务

全栈支持
推理框架

AISE 系统方法论
企业大模型战略
落地指南

国产信创
操作系统/
硬件/NPU

AISE

v0

v1

v2

2024年

2023年

当前进展

企业数字化基座

DevOpsChina.org

源于社区 服务社区

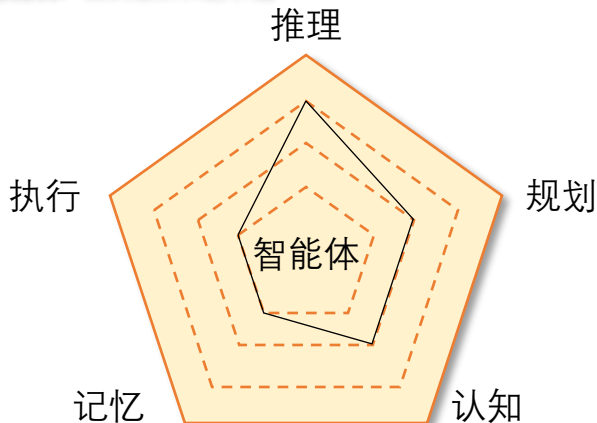
AISE

智能体能力矩阵

- ✓ **OpenAI支持**: Azure OpenAI 模型全栈支持
- ✓ **DeepSeek 全栈支持**: v1 本地化部署全栈支持, v2 MOE模型本地化部署支持, SaaS API支持
- ✓ **推理框架支持**: TGI, VLLM, MindIE
- ✓ **GPU支持**: Nvidia A100, A800, A10, L20 支持, 华为昇腾910B3, 310支持
- ✓ **国产信创**: 国产操作系统, 中间件, CPU(ARM) 支持
- **端侧推理**: 利用AI PC的本地NPU算力
- **模型微调**: 微调框架和流水线

- ✓ 已发布
- 构建中
- 规划中
- 探索中

- **能力API注册**: 为智能体提供可调用的API注册表, 允许模型使用这些API实现具体功能。
- **函数调用**: 利用模型 function call 能力, 与智能体规划能力打通
- **弹性运行时引擎**: 为智能体运行程序提供可自动弹性伸缩的计算和运行资源
- **人工介入UX**: 允许人工介入智能体运行过程, 提供简单易用的UX, 使能人与AI的流畅交互



- ✓ **提示工程**: 代码解释、评审、测试、注释、检查支持
- ✓ **提示工程**: 提示词仓库, 自助创建和分享快捷指令
- ✓ **角色定义**: 支持使用系统消息为模型指定角色
- **人工任务编排**: 提供可视化任务编排工具, 支持手工编排多阶段、多步骤复杂任务
- **函数调用**: 利用模型的 function call 能力, 自主完成多步骤任务的识别和编排
- **智能体开发框架**: 引入成熟智能体框架, 实现多角色智能体互动和自动规划能力

- ✓ **短期记忆**: 多轮对话支持 (固定历史记录长度), token限制控制
- **长期记忆**: 对话数据持久化和向量化, 支持检索和查询
- **智能记忆**: 根据当前上下文获取相关历史和上下文
- **人物画像**: 根据用户对话历史, 构建人物画像提供个性化内容生成

- ✓ **嵌入模型**: 基于OpenAI, JinnaAI 的向量化模型
- **向量数据库**: Milvus
- ✓ **文档对话**: 提供文档上传, 自动切片向量索引和基于文档内容的对话能力
- **可扩展的数据连接器**: 在内置的文件、web、jira、confluence、Github issues之外允许扩展自定义连接器, 比如: 微信网盘连接器
- ✓ **文件和跨文件认知**: SmartCode 代码生成器, AST代码结构分析
- **项目级认知**: Workspace 代码库索引、代码文本双向向量检索、代码库语义摘要生成等
- **Graph构建和检索辅助**: 利用GraphRAG增强数据检索和召回效率和准确率





Q&A



公众号：数字共生
与数字世界共生进化



公众号：DevOps
企业数字化转型实践



徐磊
北京 海淀



扫一扫上面的二维码图案，加我为朋友。



源于社区 服务社区

THANKS!



公众号：数字共生
与数字世界共生进化



公众号：DevOps
企业数字化转型实践



中国DevOps社区 2024 · 上海

