aiXcoder

AI+

基于大模型的军工创新实践

北京硅心科技有限公司

www.aiXcoder.com

源自北大

孵化自北京大学软件研究所

北京大学软件研究团队60年来一直是软件研究领域的引领者

aiXcoder是北大团队在智能化软件开发方向的重要布局

杨芙清在前苏联莫斯科大学师从舒拉波拉教授学习程序设计自动化。

团队介绍

李戈 创始人

- ✓北京大学计算机学院博雅特聘教授
- ✓教育部长江学者
- ✓中国计算机学会(CCF)软件工程专委会副主任
- ✓斯坦福大学AI实验室吴恩达团队访问教授
- ✓基于深度学习程序分析与生成领域的奠基者
- ✓科技部基金委智能化软件开发领域项目带头人

谢涛 联合创始人&首席科学家

- ✓北京大学计算机学院讲席教授、高可信软件技术教育部重点实验室副主任
- ✓UIUC终身教授
- ✓欧洲科学院外籍院士
- ✓AAAS Fellow、IEEE Fellow、ACM Fellow、CCF Fellow
- ✓全球唯一同时获得ACM软件工程领域&IEEE软件工程领域杰出服务国际奖项学者

全球智能化软件研发领域最早的引领者

aiXcoder

Known as the 1st paper on DL based Program Representation

arXiv > cs > arXiv:1409.3358

Search... Help | Advanced

Computer Science > Software Engineering

[Submitted on 11 Sep 2014]

Building Program Vector Representations for Deep Learning

Lili Mou, Ge Li, Yuxuan Liu, Hao Peng, Zhi Jin, Yan Xu, Lu Zhang

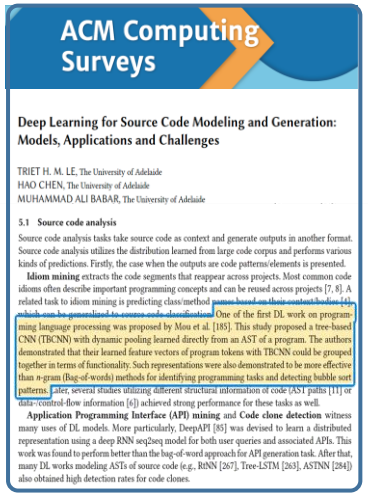
Deep learning has made significant breakthroughs in various fields of artificial intelligence. Advantages of deep learning include the ability to capture highly complicated features, weak involvement of human engineering, etc. However, it is still virtually impossible to use deep learning to analyze programs since deep architectures cannot be trained effectively with pure back propagation. In this pioneering paper, we propose the "coding criterion" to build program vector representations, which are the premise of deep learning for program analysis. Our representation learning approach directly makes deep learning a reality in this new field. We evaluate the learned vector representations both qualitatively and quantitatively. We conclude, based on the experiments, the coding criterion is successful in building program representations. To evaluate whether deep learning is beneficial for program analysis, we feed the representations to deep neural networks, and achieve higher accuracy in the program classification task than "shallow" methods, such as logistic regression and the support vector machine. This result confirms the feasibility of deep learning to analyze programs. It also gives primary evidence of its success in this new field. We believe deep learning will become an outstanding technique for program analysis in the near future.

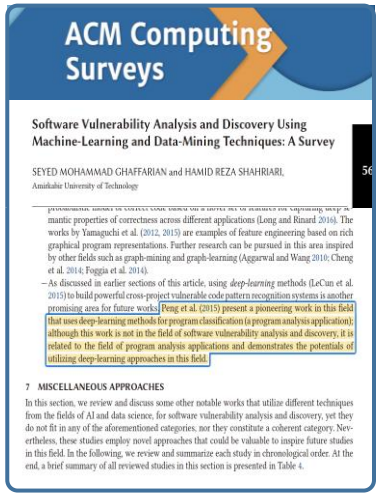
Comments: This paper was submitted to ICSE'14
Subjects: **Software Engineering (cs.SE)**; Machine Learning (cs.LG); Neural and Evolutionary Computing (cs.NE)
Cite as: arXiv:1409.3358 **[cs.SE]**
(or arXiv:1409.3358v1 **[cs.SE]** for this version)
https://doi.org/10.48550/arXiv.1409.3358

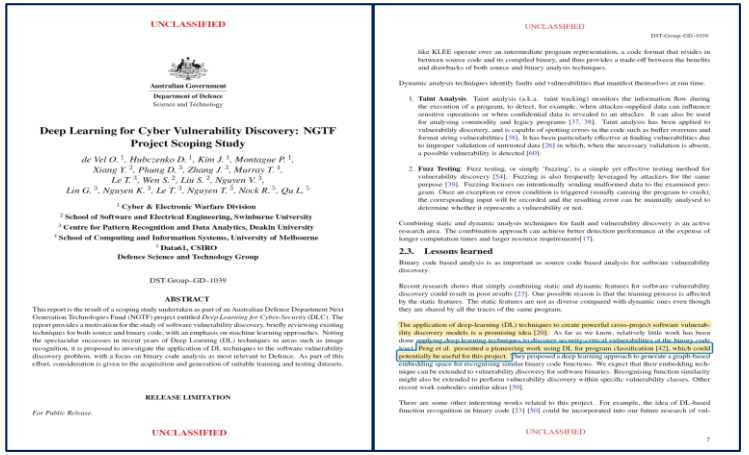
Submission history

From: Lili Mou [view email]
[v1] Thu, 11 Sep 2014 08:44:28 UTC (189 KB)

11 Sep. 2014







Lili Mou, Ge Li, Yuxuan Liu, Hao Peng, Zhi Jin, Yan Xu, Lu Zhang,
Building Program Vector Representations for Deep Learning. arXiv preprint arXiv:1409.3358, 2014.

Known as the 1st paper on DL based Program Representation

Computer Science > Machine Learning

[Submitted on 18 Sep 2014 (v1), last revised 8 Dec 2015 (this version, v2)]

Convolutional Neural Networks over Tree Structures for Programming Language Processing

Lili Mou, Ge Li, Lu Zhang, Tao Wang, Zhi Jin

Programming language processing (similar to natural language processing) is a hot research topic in the field of software engineering; it has also aroused growing interest in the artificial intelligence community. However, different from a natural language sentence, a program contains rich, explicit, and complicated structural information. Hence, traditional NLP models may be inappropriate for programs. In this paper, we propose a novel tree-based convolutional neural network (TBCNN) for programming language processing, in which a convolution kernel is designed over programs' abstract syntax trees to capture structural information. TBCNN is a generic architecture for programming language processing; our experiments show its effectiveness in two different program analysis tasks: classifying programs according to functionality, and detecting code snippets of certain patterns. TBCNN outperforms baseline methods, including several neural models for NLP.

Comments: Accepted at AAAI-16
Subjects: **Machine Learning (cs.LG)**; Neural and Evolutionary Computing (cs.NE); Software Engineering (cs.SE)
Cite as: arXiv:1409.5718 **[cs.LG]**
(or arXiv:1409.5718v2 **[cs.LG]** for this version)
https://doi.org/10.48550/arXiv.1409.5718

Submission history
From: Lili Mou [view email]
[v1] Thu, 18 Sep 2014 06:50:52 UTC (220 KB)
[v2] Tue, 8 Dec 2015 12:31:51 UTC (310 KB)

18 Sep. 2014

ACM Computing Surveys

Deep Learning for Source Code Modeling and Generation: Models, Applications and Challenges

TRIET H. M. LE, The University of Adelaide
HAO CHEN, The University of Adelaide
MUHAMMAD ALI BABAR, The University of Adelaide

5.1 Source code analysis

Source code analysis tasks take source code as context and generate outputs in another format. Source code analysis utilizes the distribution learned from large code corpus and performs various kinds of predictions. Firstly, the case when the outputs are code patterns/elements is presented.

Idiom mining extracts the code segments that reappear across projects. Most common code idioms often describe important programming concepts and can be reused across projects [7, 8]. A related task to idiom mining is predicting class/method names based on their context/bodies [4], which can be generalized to source code classification. One of the first DL work on programming language processing was proposed by Mou et al. [185]. This study proposed a tree-based CNN (TBCNN) with dynamic pooling learned directly from an AST of a program. The authors demonstrated that their learned feature vectors of program tokens with TBCNN could be grouped together in terms of functionality. Such representations were also demonstrated to be more effective than *n*-gram (Bag-of-words) methods for identifying programming tasks and detecting bubble sort patterns. Later, several studies utilizing different structural information of code (AST paths [11] or data-/control-flow information [6]) achieved strong performance for these tasks as well.

Application Programming Interface (API) mining and **Code clone detection** witness many uses of DL models. More particularly, DeepAPI [85] was devised to learn a distributed representation using a deep RNN seq2seq model for both user queries and associated APIs. This work was found to perform better than the bag-of-word approach for API generation task. After that, many DL works modeling ASTs of source code (e.g., RtNN [267], Tree-LSTM [263], ASTNN [284]) also obtained high detection rates for code clones.

Lili Mou, Ge Li, Lu Zhang, Tao Wang, Zhi Jin, Convolutional Neural Networks over Tree Structures for Programming Language Processing, arXiv preprint arXiv:1409.5718, 2014.

全球智能化软件研发领域最早的引领者

aiXcoder

Known as the 1st paper on DL based Code Generation

arXiv > cs > arXiv:1510.07211

Search... Help | Advanced

Computer Science > Software Engineering

[Submitted on 25 Oct 2015]

On End-to-End Program Generation from User Intention by Deep Neural Networks

Lili Mou, Rui Men, Ge Li, Lu Zhang, Zhi Jin

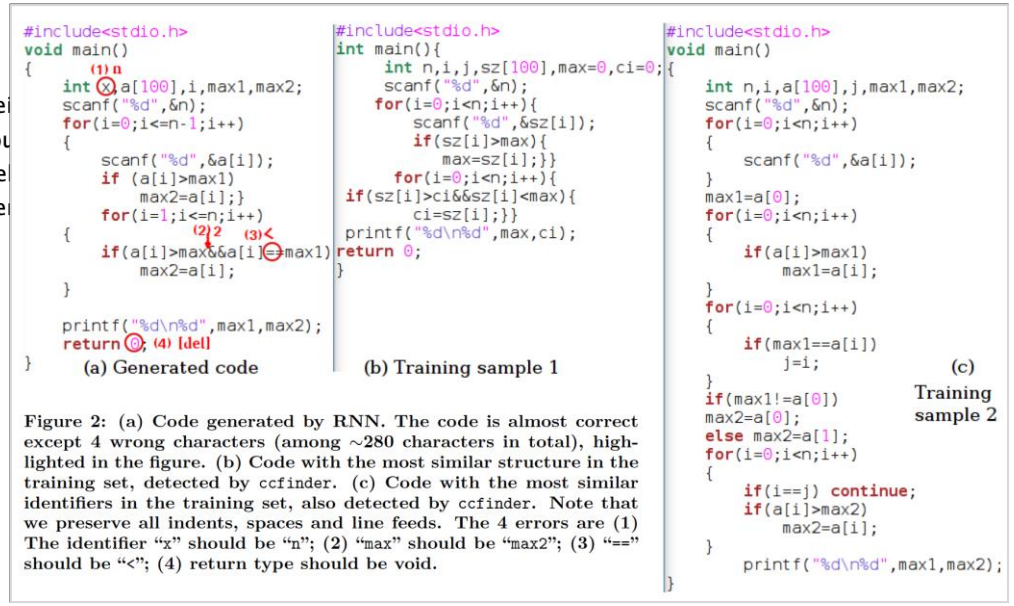
This paper envisions an end-to-end program generation scenario using recurrent neural networks (RNNs): Users can express thei
RNN then automatically generates corresponding code in a characterby-by-character fashion. We demonstrate its feasibility throu
analysis. To fully make such technique useful in practice, we also point out several cross-disciplinary challenges, including mode
improving model architectures, etc. Although much long-term research shall be addressed in this new field, we believe end-to-e
a reality in future decades, and we are looking forward to its practice.

Comments: Submitted to 2016 International Conference of Software Engineering "Vision of 2025 and Beyond" track
Subjects: **Software Engineering (cs.SE)**; Machine Learning (cs.LG)
Cite as: arXiv:1510.07211 **[cs.SE]**
(or arXiv:1510.07211v1 **[cs.SE]** for this version)
https://doi.org/10.48550/arXiv.1510.07211

Submission history

From: Lili Mou [view email]
[v1] Sun, 25 Oct 2015 06:52:45 UTC (217 KB)

25 Oct. 2015



```
#include<stdio.h>
void main()
{
    int x,a[100],i,max1,max2;
    scanf("%d",&n);
    for(i=0;i<=n-1;i++)
    {
        scanf("%d",&a[i]);
        if (a[i]>max1)
            max2=a[i];}
        for(i=1;i<=n;i++)
    {
        if(a[i]>max&&a[i]==max1)
            max2=a[i];
    }
    printf("%d\n%d",max1,max2);
    return 0;
}
```

(a) Generated code

```
#include<stdio.h>
int main(){
    int n,i,j,sz[100],max=0,ci=0;
    scanf("%d",&n);
    for(i=0;i<n;i++){
        scanf("%d",&sz[i]);
        if(sz[i]>max){
            max=sz[i];}}
    for(i=0;i<n;i++){
if(sz[i]>ci&&sz[i]<max){
        ci=sz[i];}}
printf("%d\n%d",max,ci);
return 0;
}
```

(b) Training sample 1

```
#include<stdio.h>
void main()
{
    int n,i,a[100],j,max1,max2;
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    max1=a[0];
    for(i=0;i<n;i++)
    {
        if(a[i]>max1)
            max1=a[i];
    }
    for(i=0;i<n;i++)
    {
        if(max1==a[i])
            j=i;
    }
    if(max1!=a[0])
    max2=a[0];
    else max2=a[1];
    for(i=0;i<n;i++)
    {
        if(i==j) continue;
        if(a[i]>max2)
            max2=a[i];
    }
        printf("%d\n%d",max1,max2);
}
```

(c) Training sample 2

Figure 2: (a) Code generated by RNN. The code is almost correct except 4 wrong characters (among ~280 characters in total), highlighted in the figure. (b) Code with the most similar structure in the training set, detected by `ccfinder`. (c) Code with the most similar identifiers in the training set, also detected by `ccfinder`. Note that we preserve all indents, spaces and line feeds. The 4 errors are (1) The identifier "x" should be "n"; (2) "max" should be "max2"; (3) "==" should be "<"; (4) return type should be void.

Lili Mou, Rui Men, Ge Li, Lu Zhang, Zhi Jin, On End-to-End Program Generation from User Intention by Deep Neural Networks, arXiv preprint, arXiv:1510.07211, 2015.

坚实的科研基础

Applications in Industry

Task Oriented Researches

Task Independent Researches

[ICPC 2023] ACM SIGSOFT Distinguished Paper Award.

[NeurIPS 2022] Big Model Training.

[NIPS 2021] Code Transformer Representation.

[TOSEM, 2021] the First Paper on DL Robustness.

[TOSEM, 2020] Modular base Code Representation.

[ASE 2020] the First Paper on Pre-training based Code Completion.

[NIPS 2019] Popular Cited Paper on Code Generation and Summarization.

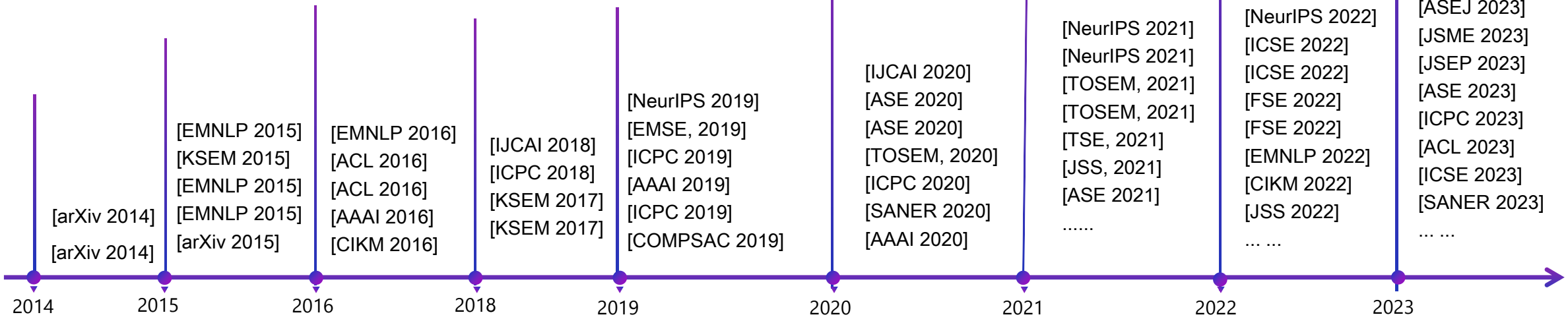
[IJCAI 2018] Popular Cited Paper on Program Summarization.

[ICPC 2018] Popular Cited Paper on Program Summarization.

[AAAI 2016] the First Paper on AST based Program Representation.

[arXiv 2015] the First DL Work on Program Generation.

[arXiv 2014] the First DL Work on Program Representation.



长期致力于知识检索、知识工程研究

专 利

序号	专利名称	取得日期
1	分类知识获取方法和装置	2016年
2	一种多维度领域关键知识的提取和存储方法	2019年
3	以关系为核心的层次化知识建模结构及知识图谱构建方法	2022年
4	面向基因数据的联邦分析系统和方法、设备及介质	2022年
5	信息检索知识图谱嵌入方法、装置和计算机设备	2023年

&

论 文

序号	论文标题	发表日期
1	一种面向主题的Web知识检索方法	2012年
2	基于Web的知识工程	2012年
3	基于增强描述的代码搜索方法	2017年
4	面向医疗问答系统的大语言模型命名实体识别方法	2023年
5	ChatModeler：基于大语言模型的人机协作迭代式需求获取和建模方法	2024年

AIXcoder

深耕大模型

产品迭代路线图

aiXcoder

2013

开始把深度学习应用于程序分析研究

2014.9

第一篇
基于DL的代码表示论文

2015.10

第一篇
基于DL的代码生成论文

2017.5

原型版 0.1

- ✓ 语言：Python
- ✓ IDE：Eclipse

2018.6

aiXcoder 0.6

- ✓ 语言：Java, Android
- ✓ IDE：IntelliJ IDEA

2018.12

aiXcoder 0.8

- ✓ Python
- ✓ IDE: PyCharm

2019

aiXcoder1.0上线

- ✓ 在线版
- ✓ 代码智能补全&搜索

2020

aiXcoder2.0上线

- ✓ 本地版
- ✓ 代码补全引擎

2021.3

aiXcoder3.0上线

- ✓ 支持本地个性化训练
- ✓ 支持语言种类大幅度完善

2022.4

aiXcoderL上线

- ✓ 十亿级大模型，支持代码补全和自然语言推荐
- ✓ 是国内首个基于“大模型”的智能编程商用产品

2022.6

aiXcoder XL上线

- ✓ 推出百亿级亿参数大模型，国内首个完全自主知识产权的方法级代码生成模型
- ✓ 支持自然语言生成方法级代码，支持多行补全、Token补全
- ✓ 代码生成能力超过业界最佳-GitHub Copilot
- ✓ 支持企业私有化部署
- ✓ 可对企业代码进行个性化训练

2023.8

aiXcoder Europa上线

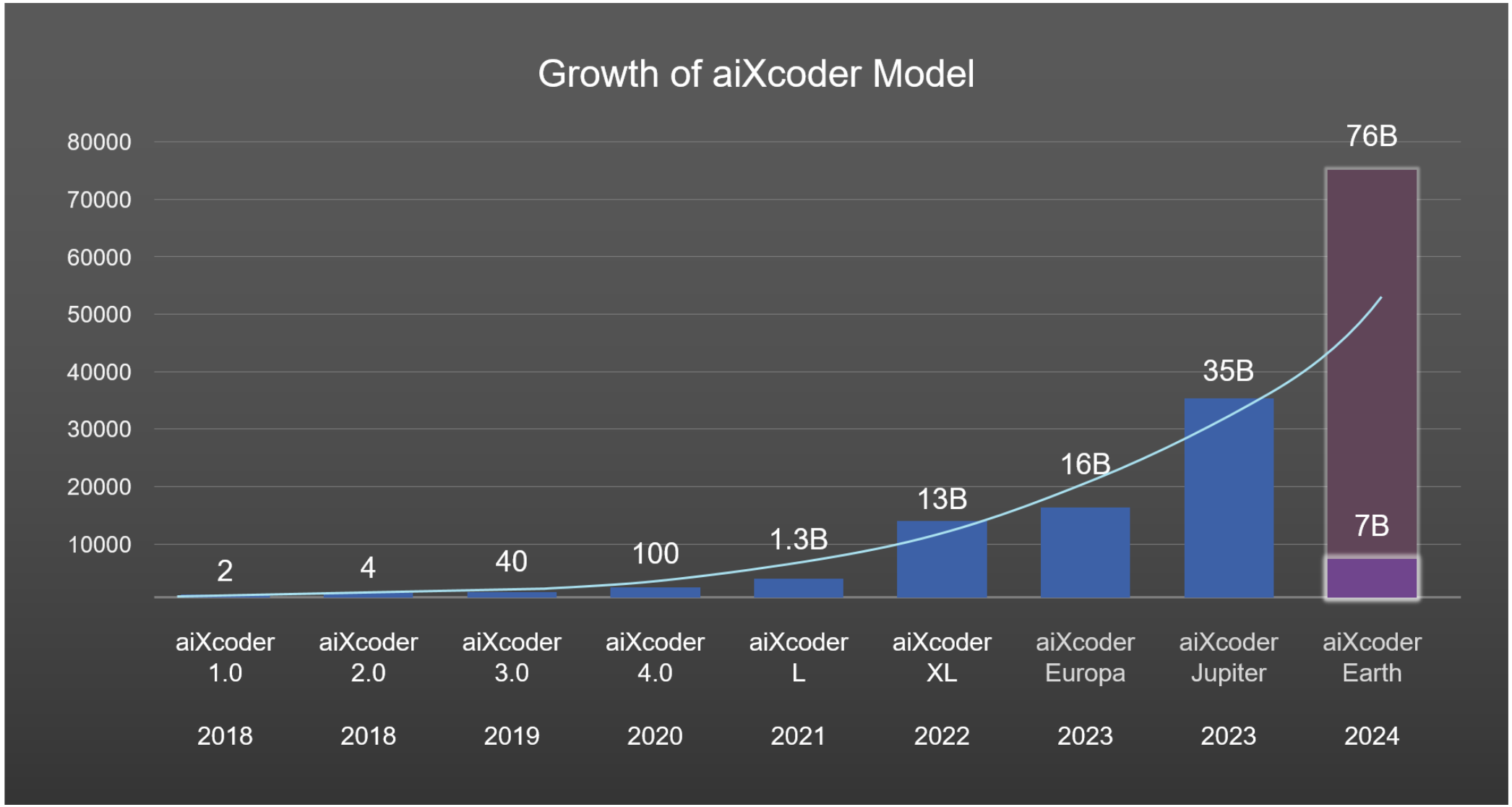
- ✓ 单元测试代码自动生成
- ✓ 代码缺陷检测与修复
- ✓ 代码注释自动生成
- ✓ 代码解释

2024

aiXcoder 76B/7B

- ✓ 适用于工业级软件研发的多场景智能化研发平台
- ✓ 领域模型优化
- ✓ 模型规模扩大
- ✓ 模型性能增强

aiXcoder大模型迭代



aiXcoder

aiXcoder XL

支持方法级代码生成

根据自然语言功能描述，生成完整程序代码

对标 Copilot，国内首个：自然语言一键生成方法级代码 aiXcoder XL…

国内首个能够根据开发者给出的“自然语言描述”一键生成“完整方法级代码”的智能编程应…

机器之心

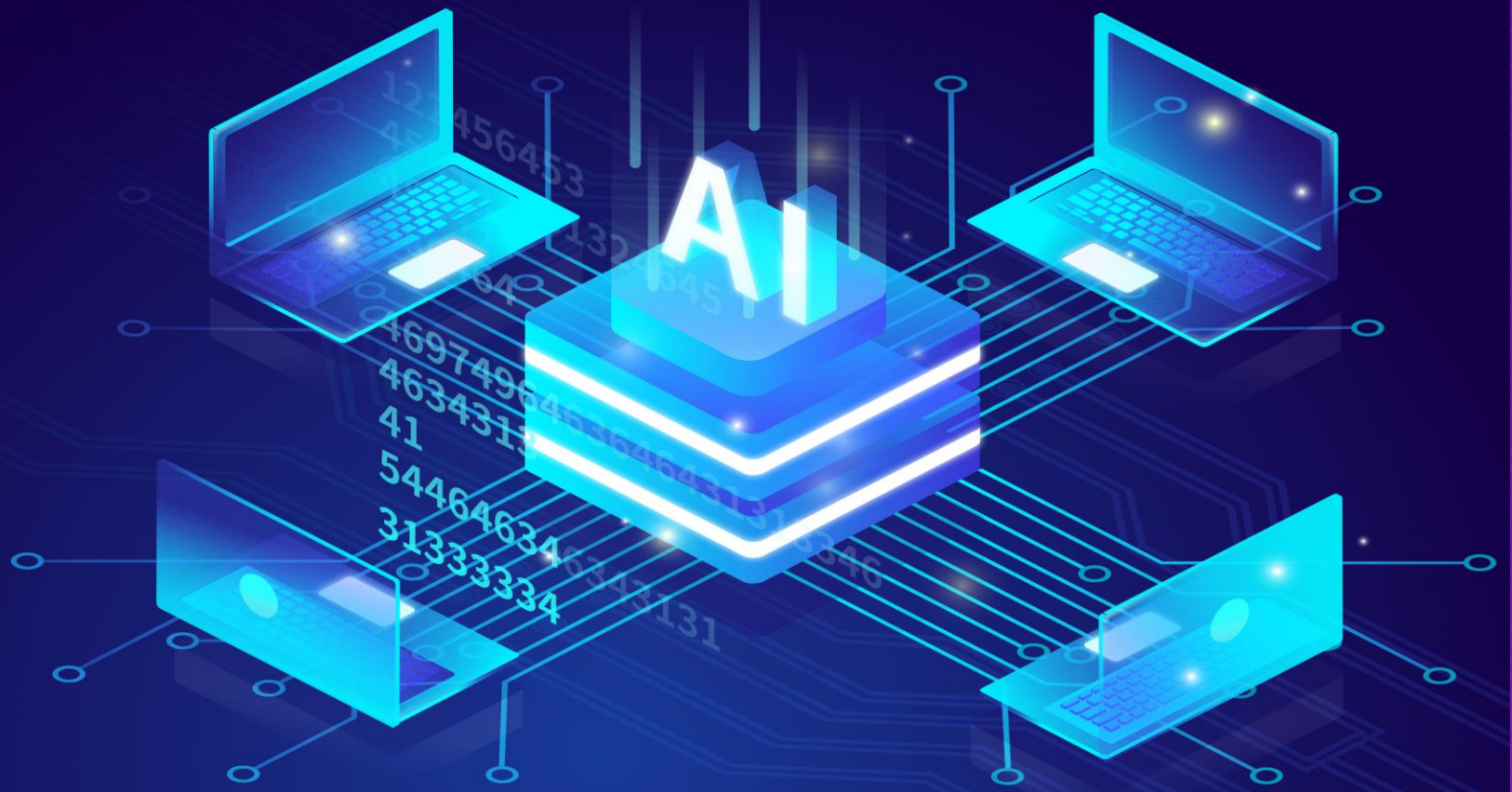
国内首个！ 2022年6月22日

2023年8月， aiXcoder 全新版本发布

新增4大功能：

- 单元测试代码自动生成
- 代码缺陷检测与修复
- 代码注释自动生成
- 代码解释

覆盖更多开发场景



aiXcoder开源7B大模型

2024年04月

aiXcoder开源7B大模型

aiXcoder

Capability = Model Prior + Training Data

aiXcoder 7B坚持自主研发，采用更新的深度学习架构，并使用超过1.2T（1.2万亿） Unique Token 高质量数据进行训练，全球单批次训练最大数据量。

- ✓ 生成速度快：<20 ms/token
- ✓ 部署成本低：一张V100
- ✓ 并发能力强：单张A100部署4个模型
- ✓ 信创适配: 昇腾910

Language	Num. Files(M)	Volume (GB)	Filtered Num. Files(M)	Filtered Volume (GB)
java	526.5	1837.4	209.2	648.21
c++	671.2	1826.1	127.1	326.56
python	202.5	925.3	81.9	353.93
javascript	459.1	4816	148.7	1281.1
go	207	298	98.6	120.3
typescript	67.3	285.2	31.5	91
c#	145.4	769.7	63.6	262.3
kotlin	6.5	13.2	3.2	6.7

Table 10 | The amount of data in main language before filtering and after filtering

aiXcoder开源7B大模型

- ✓ aiXcoder-7B模型与各大百亿级参数量预训练基础模型相比，在主流NL2Code 基准上表现最好
- ✓ 值得关注的是，aiXcoder-7B 模型效果好于5倍参数量的CodeLlama-34B

Model	HumanEval	MBPP	MultiPL-E-C++	MultiPL-E-Java	MultiPL-E-JS	Avg
CodeGen2.5-7B	28.7%	39.2%	25.7%	26.1%	26.2%	29.1%
CodeGeex2-7B	36.0%	36.2%	29.2%	25.9%	24.8%	30.4%
CodeLlama-7B	31.7%	38.6%	29.8%	34.2%	29.2%	32.7%
CodeShell-7B	34.4%	38.6%	28.2%	30.4%	33.2%	32.9%
StarCoder2-7B	35.4%	54.4%	33.6%	29.4%	35.4%	37.6%
DeepSeekCoder-6.7B	49.4%	60.6%	50.3%	43.0%	48.4%	50.3%
aiXcoder-7B	**54.9%**	**66.0%**	**58.2%**	**57.0%**	**64.5%**	**60.1%**
StarCoder-15B	31.7%	42.8%	31.1%	28.5%	29.8%	32.8%
CodeLlama-13B	36.0%	48.4%	37.9%	38.0%	32.3%	38.5%
StarCoder2-15B	46.3%	66.2%	41.4%	33.9%	44.2%	46.4%
CodeLlama-34B	48.2%	55.2%	44.7%	44.9%	42.2%	47.0%

Performance(Pass@1) on HumanEval, MBPP and MultiPL-E Benchmarks.

代码补全能力 – 研发最关注场景

- ✓ **与上面表格的**Stand alone nl2code **不同，在实际编程场景中，我们更关注代码补全。**
- ✓ **主流评测集是**Santacoder(Ben Allal et al., 2023)，**该评测集会从**HumanEval **或者**MultiPL-E **中按行随机抽取代码，然后在给定完整的上文与下文条件下，评估模型生成结果的**Exact Match**指标。**

Model	Python	JavaScript	Java	Avg
StarCoder2-7B	61.1%	77.5%	81.1%	73.2%
CodeLlama-7B	67.6%	74.3%	80.2%	74.0%
CodeLlama-13B	68.3%	77.6%	80.7%	75.5%
DeepSeekCoder-7B	66.6%	79.7%	**88.1%**	78.1%
aiXcoder-7B-Base	**73.3%**	**81.7%**	83.0%	**79.3%**

Performance(Exact Match) on single-line infilling on the FIM benchmark by Ben Allal et al.

* 代码补全能力 – 真实开发场景（aiXcoder开源测试集）

「 aiXcoder 2024年新公布的开源测试集 」，贴合实际开发场景：数据量更大，被测代码多样性更高、被测代码上下文长度更长、是更接近实际开发项目的评测集。

Model	Java			
	Exact Match	BLEU-4	CodeBLEU	Length(Pred/Ref)
CodeLLama-7B	38.1	56.9	69.9	340/159
StarCoder2-7B	37.7	57.7	69.2	353/159
DeepSeekCoder-7B	43.4	63.4	71.7	217/159
aiXcoder-7B-Base	**49.4**	**70.6**	**74.0**	154/159

Model	C++			
	Exact Match	BLEU-4	CodeBLEU	Length(Pred/Ref)
CodeLLama-7B	25.4	44.2	63.9	486/161
StarCoder2-7B	22.6	41.9	61.2	583/161
DeepSeekCoder-7B	29.1	50.0	65.8	330/161
aiXcoder-7B-Base	**37.3**	**60.3**	**67.4**	217/161

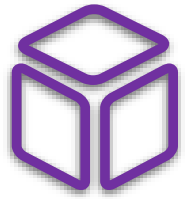
* 跨文件代码生成「 企业真实开发场景 」

aiXcoder-7B 模型在贴近真实开发场景的评测集CrossCodeEval上，aiXcoder-7B一举拿下了同级别模型的最好效果。

Model	Code Match							
	Python		Java		TypeScript		C#	
	EM	ES	EM	ES	EM	ES	EM	ES
CodeLlama-7B	22.3	55.2	27.9	66.9	10.8	70.9	45.8	77.2
+ Retrieval BM25	23.5	53.5	33.9	68.4	11.5	71.5	50.6	75.3
+ Retrieval w/Ref.	26.7	54.9	36.3	69.0	12.8	72.9	52.8	75.0
StarCoder2-7B	22.5	57.3	25.9	65.9	28.9	71.6	39.5	70.5
+ Retrieval BM25	25.3	58.0	31.4	67.4	33.3	73.2	43.5	69.8
+ Retrieval w/Ref.	28.5	59.0	35.0	69.2	36.0	72.6	47.9	71.6
DeepSeekCoder-7B	27.2	62.3	33.4	73.2	36.6	77.3	45.9	77.0
+ Retrieval BM25	29.9	62.9	39.8	74.8	39.0	77.0	52.2	78.1
+ Retrieval w/Ref.	33.2	64.5	43.7	76.1	43.4	78.4	55.4	78.7
aiXcoder-7B-Base	30.0	70.8	34.9	77.8	35.3	79.6	49.9	86.4
+ Retrieval BM25	35.3	74.3	42.2	80.4	39.9	81.3	57.7	88.8
+ Retrieval w/Ref.	40.4	76.3	47.0	82.4	45.0	83.8	61.0	89.4

Performance of various code LLMs on CrossCodeEval. "Retrieval" and "Retrieval w/ Ref." mean retrieved with the prompt and the prompt + reference we construct the prompt by prepending the retrieved cross-file context.

AIXcoder 大模型系列产品之一

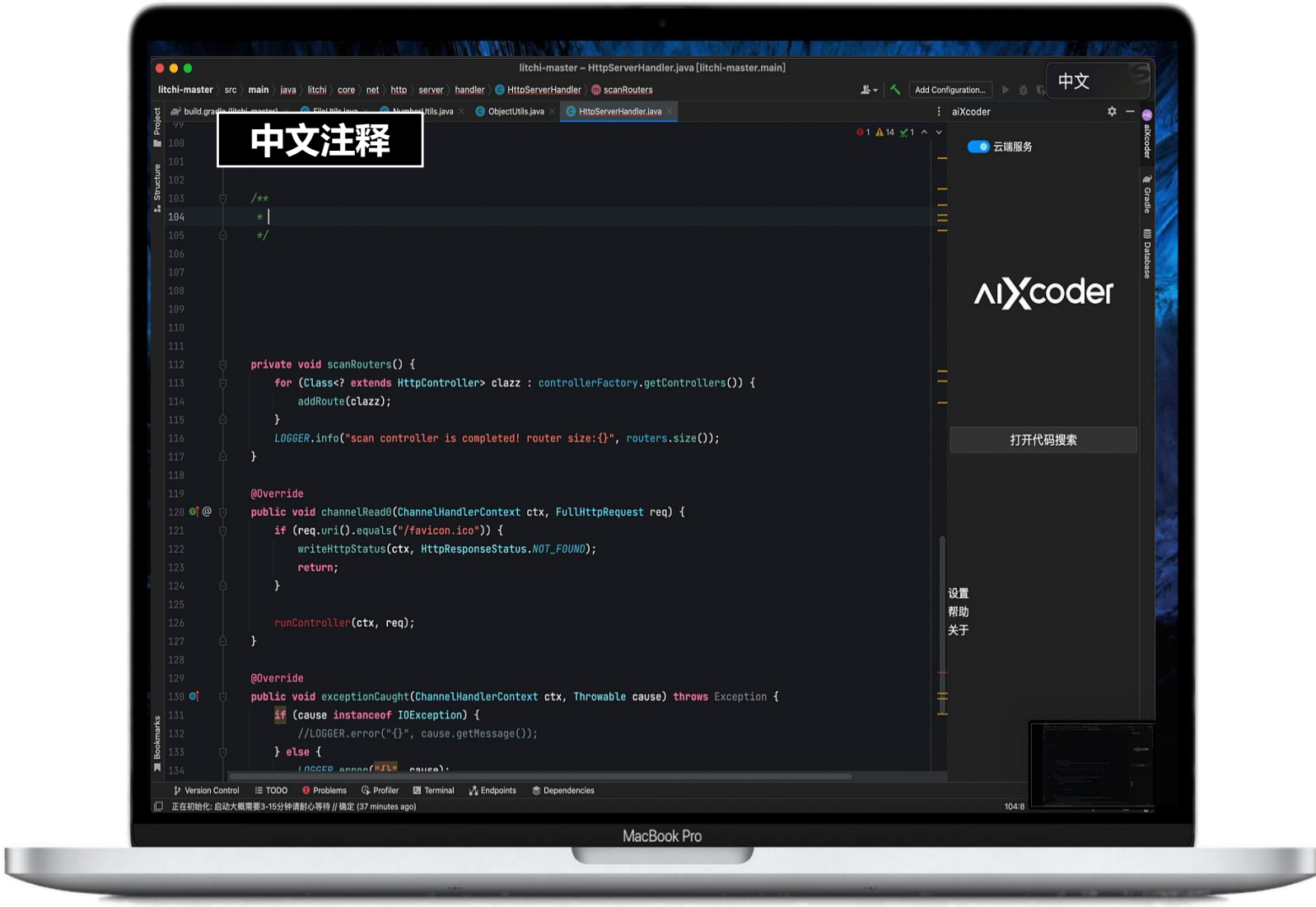


智能编程系统

智能编程系统：代码自动生成

代码自动生成 功能可以根据开发者输入的自然语言描述，结合上下文的逻辑关系来生成完整的函数级代码和注释。

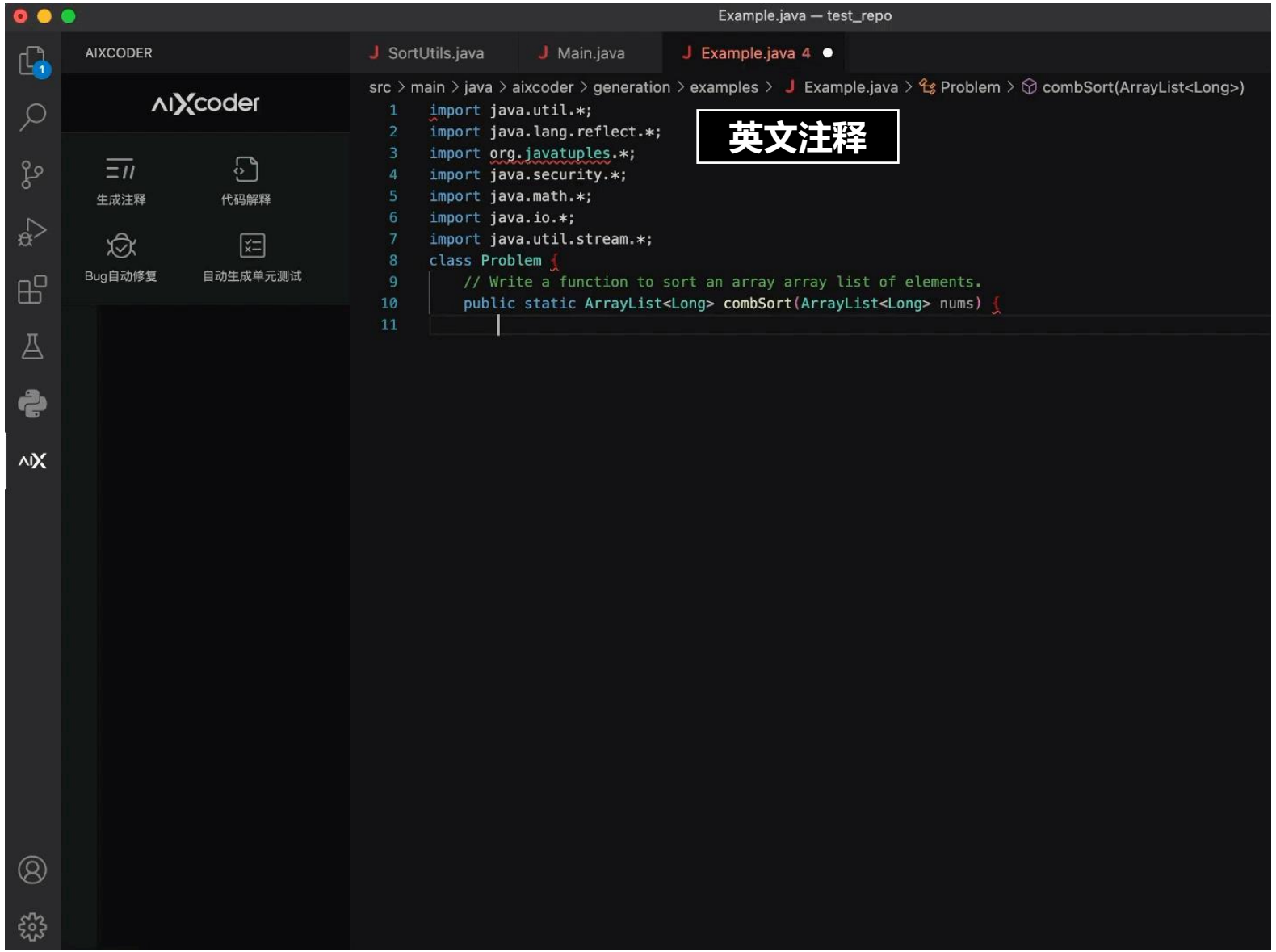
当开发者面对不熟悉的场景时，利用“代码生成”的能力，能有效帮助开发者熟悉陌生场景，提高开发效率。



智能编程系统：代码自动生成

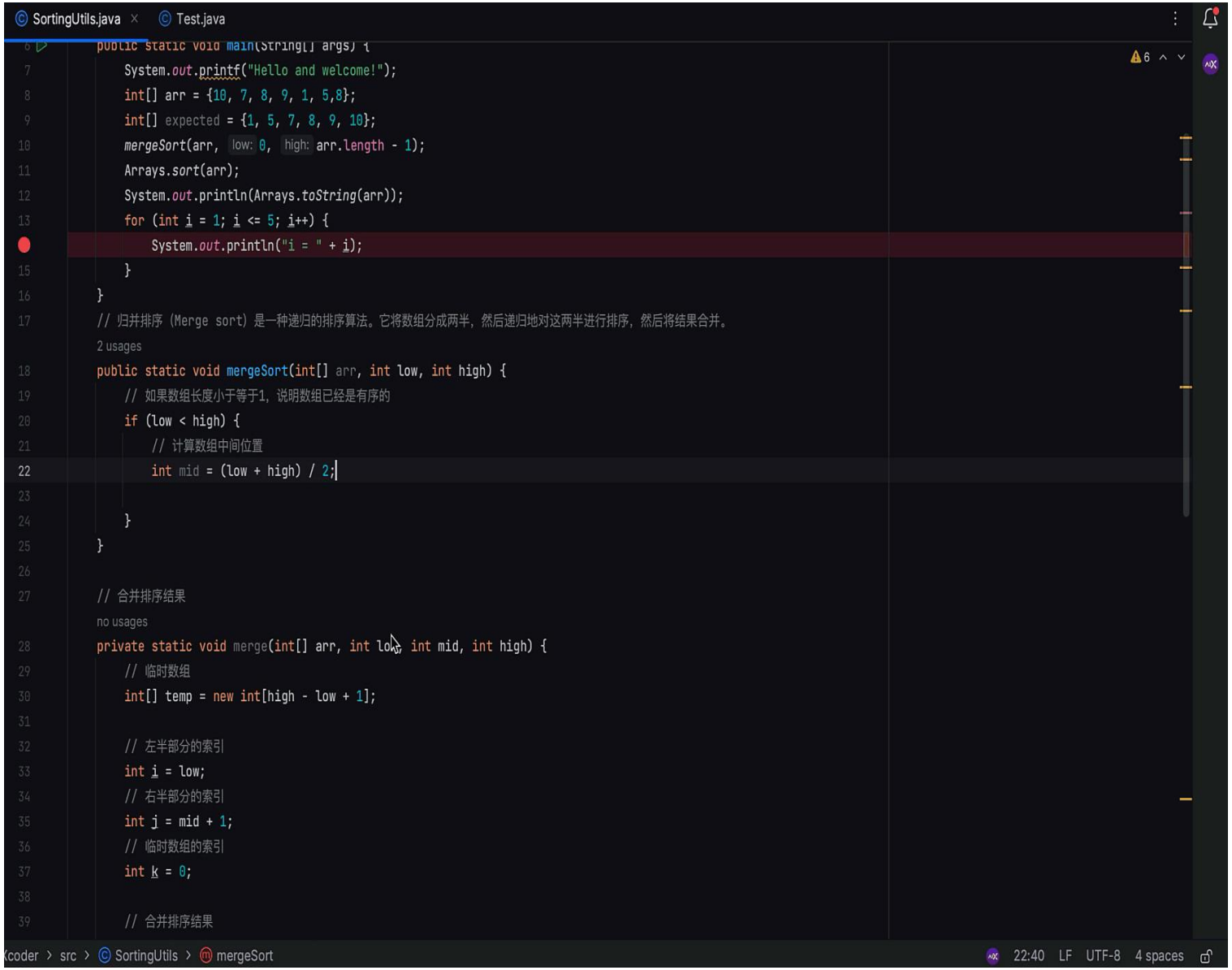
代码自动生成 功能可以根据开发者输入的自然语言描述，结合上下文的逻辑关系来生成完整的函数级代码和注释。

当开发者面对不熟悉的场景时，利用“代码生成”的能力，能有效帮助开发者熟悉陌生场景，提高开发效率。



智能编程系统：代码自动补全

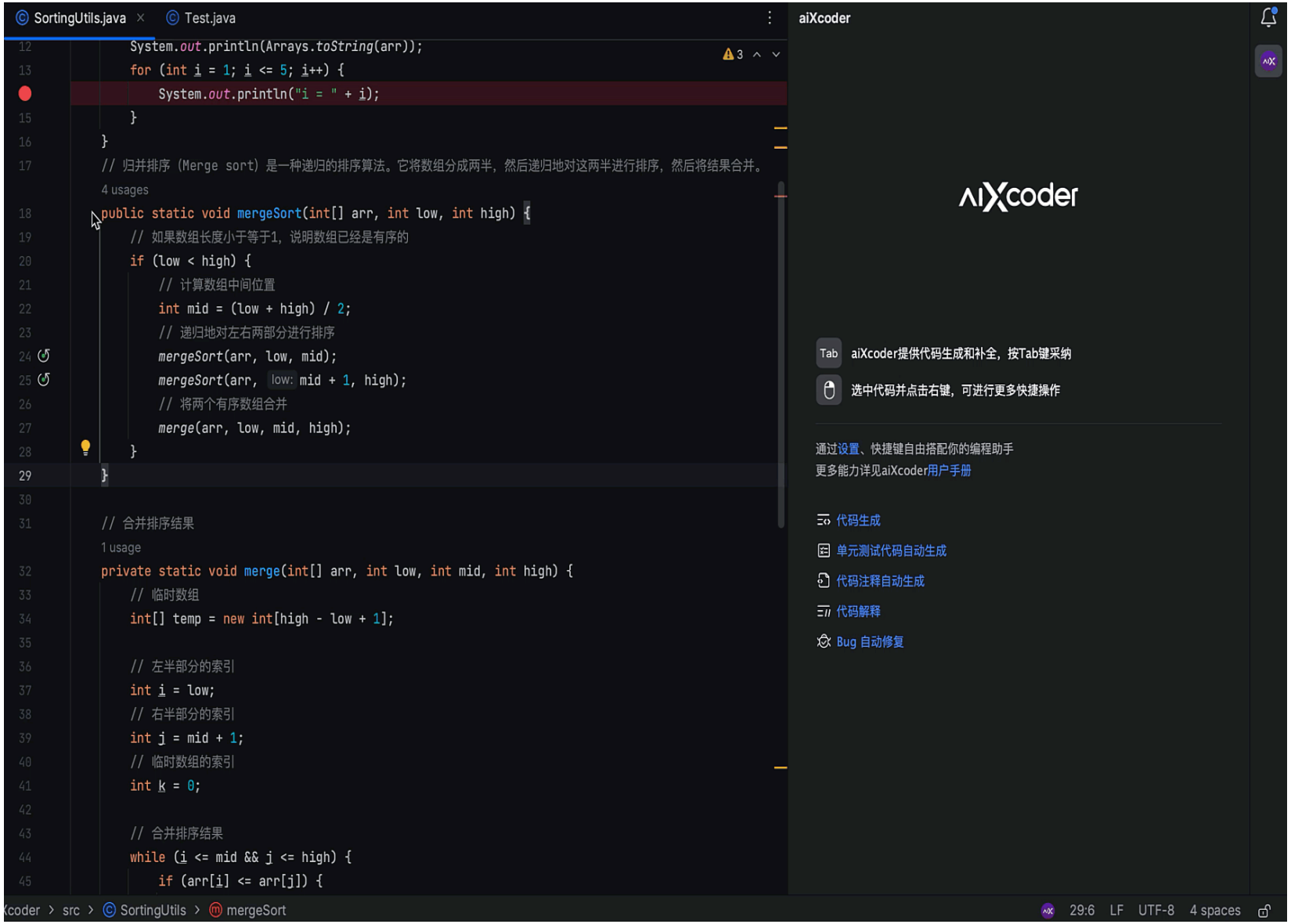
代码自动补全 功能可以通过分析文本的语法结构、语义关系、上下文逻辑关系以及同一项目下的其他文件，自动生成符合当前业务逻辑的行级、函数级代码，同时生成相应注释，单击Tab键即可轻松采纳。



智能编程系统：单元测试代码自动生成

单元测试代码自动生成 功能支持针对任意方法、函数生成单元测试用例和相关代码解释。

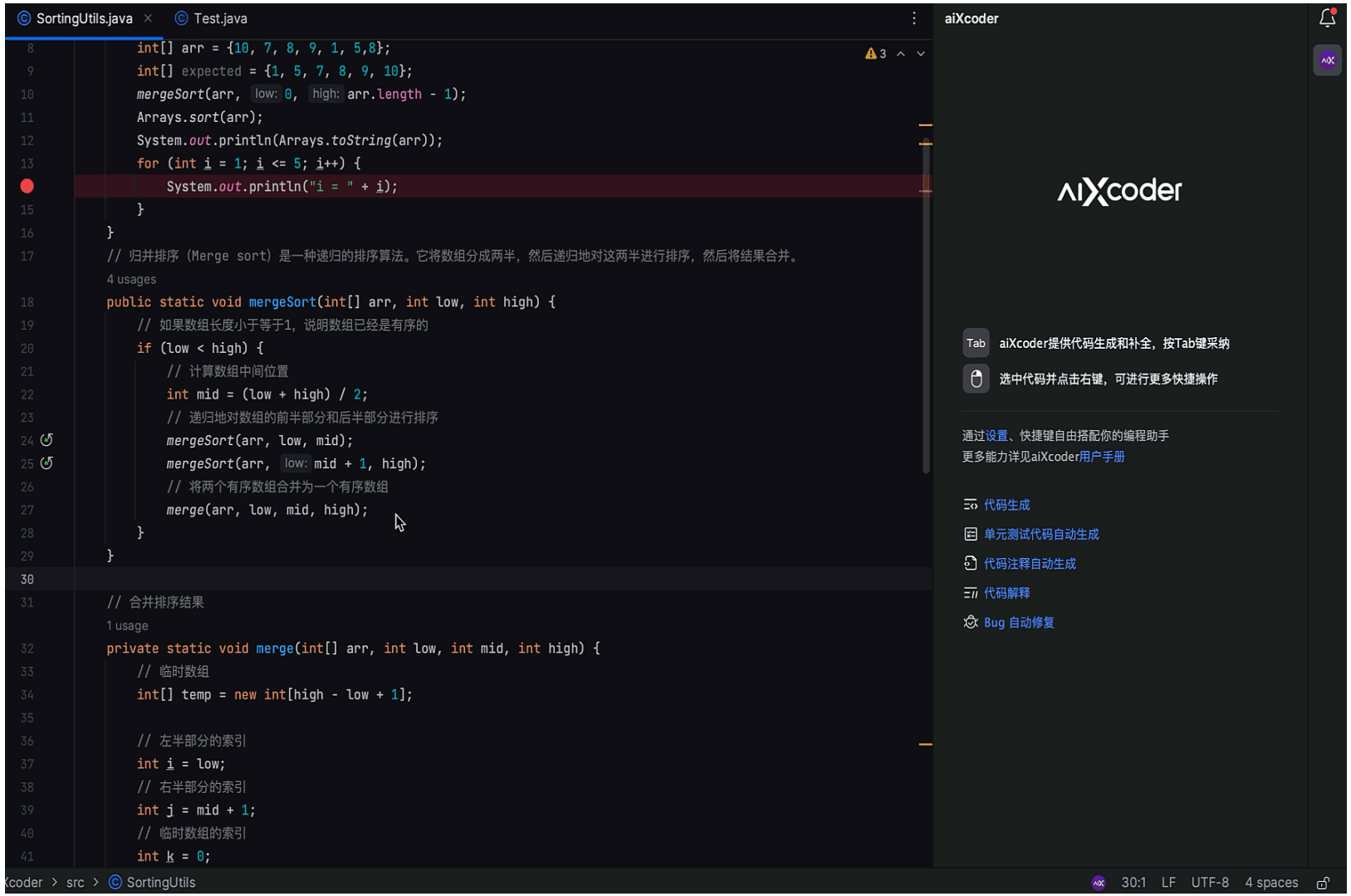
在开发者编写代码后，通过“单元测试代码自动生成”功能自动生成对应的单元测试，保障代码质量



智能编程系统：代码缺陷检测与修复

代码缺陷检测与修复 功能能够自动识别并修复代码中的错误，通过分析被测代码，识别代码中的潜在错误，并自动给出修复后的代码。

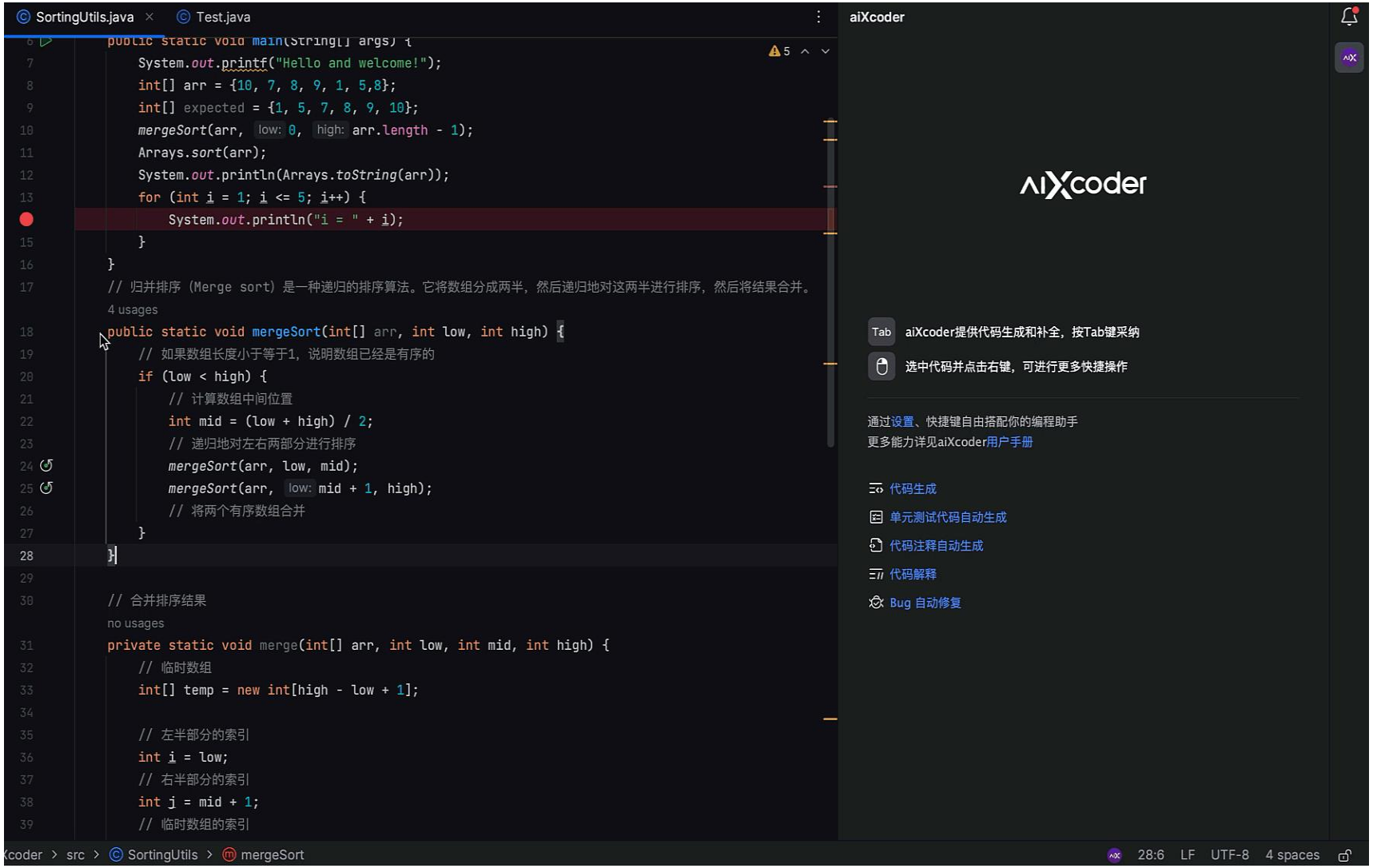
这个功能在提高代码稳定性和可靠性方面有很大的价值，同时也可以减少手动调试和修复Bug的时间



智能编程系统：代码解释

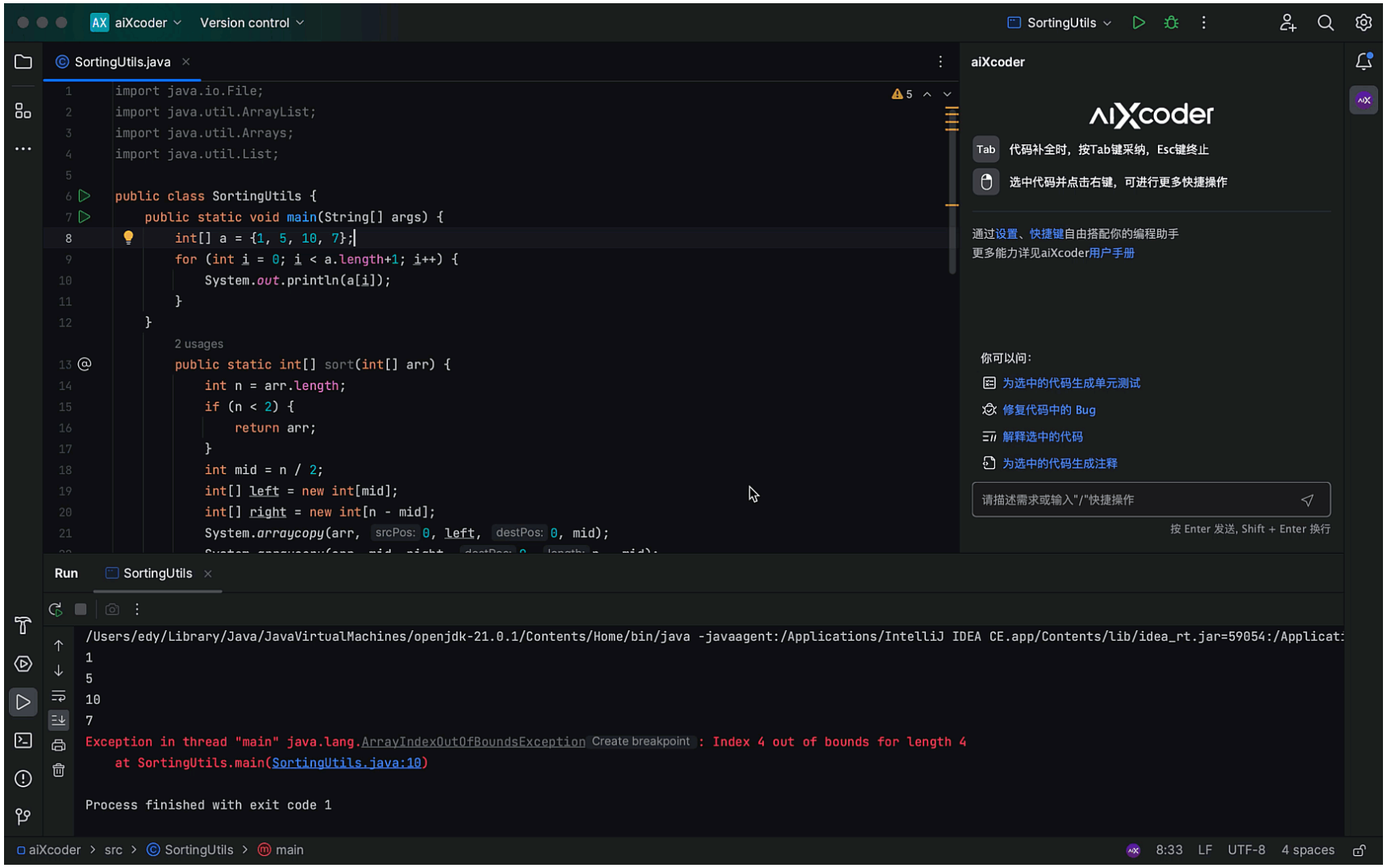
代码解释 功能支持数十种编程语言的识别，能够自动解析代码的功能和结构，从而生成对应的文本解释。

该功能可以有效减轻开发者编写功能文档的负担，同时帮助开发者快速理解代码逻辑和功能设计。



智能编程系统：代码问答

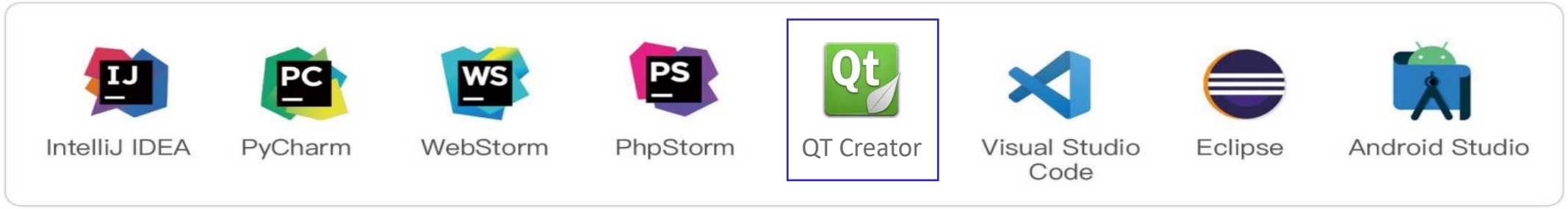
代码问答 报错信息解释， 结合报错信息对原始代码做修复



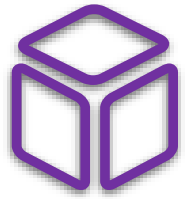
智能编程系统 - 支持多语言、多IDE

- 支持100+编程语言

- 支持多种主流和非主流IDE

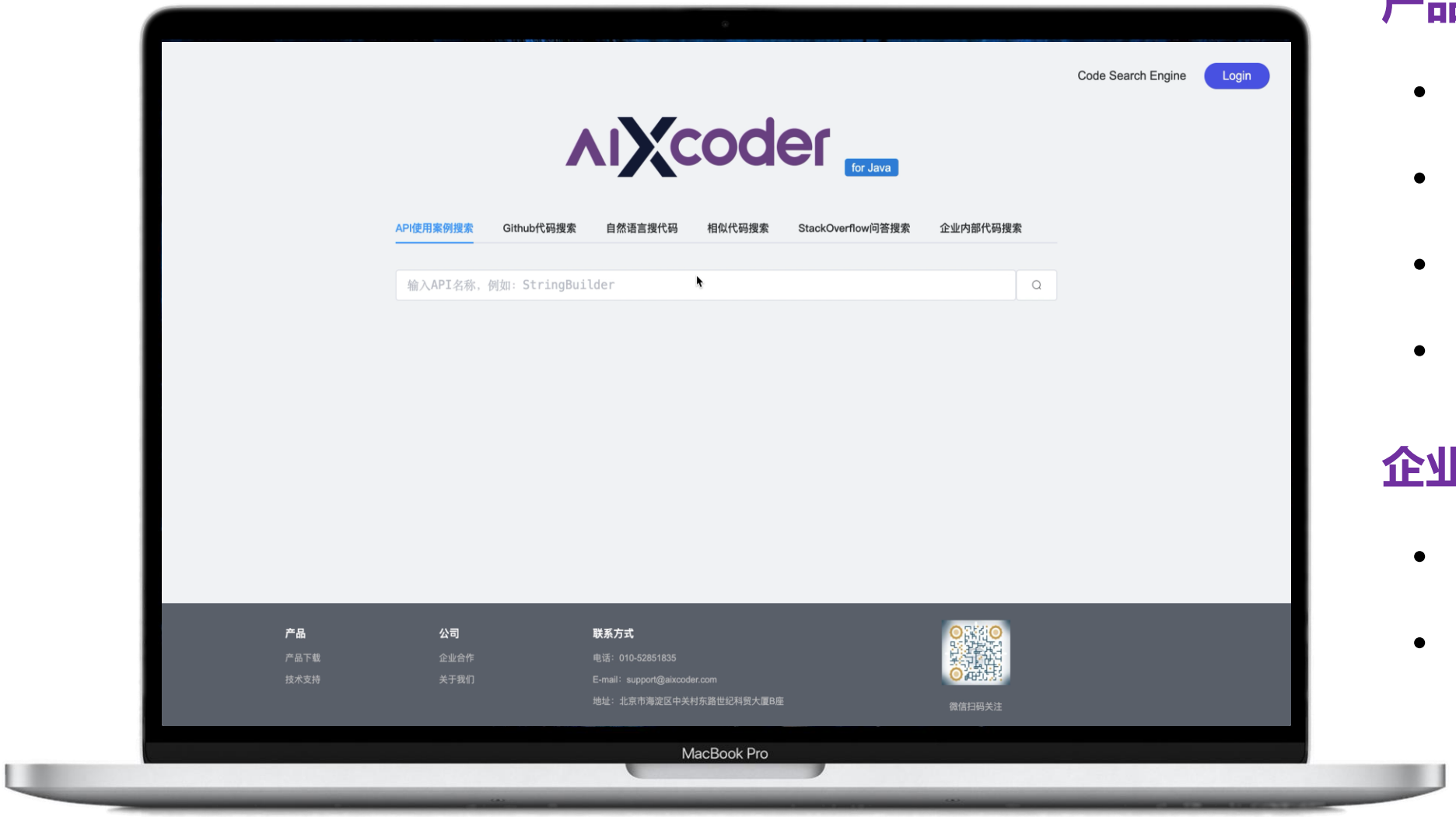


AIXcoder 大模型系列产品之二



智能搜索引擎

智能代码搜索



产品功能

- API使用案例搜索
- 自然语言搜索代码
- 代码片段搜索代码
- 常见代码问题搜索

企业服务

- 企业API案例提取
- 企业内部代码搜索

智能代码搜索

aiXcoder智能代码搜索从海量开源代码以及编程社区中获取了优秀代码样例、代码相关问题的回答，用来帮助软件开发人员所需要的搜索。尤其在访问外网受限的企业中，aiXcoder的代码搜索功能可以给软件开发人员提供很大的便利。

技术特点：

1、采用基于深度学习的代码理解技术，能够对代码语义进行精准解析。

2、支持在缺乏代码注释的情况下，根据代码功能和用途搜索相关代码。

3、对企业代码进行语义解析、缺陷扫描过滤，完成API使用案例提取。

企业服务

企业私有化部署

智能代码搜索引擎打包集成了GitHub、Gitee和StackOverFlow等网站优秀的代码和问答知识，在企业内部私有化部署后，为内外网隔离的企业开发者查询资料解决问题提供极大的便利。

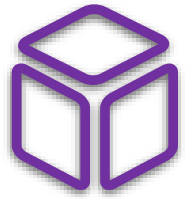
企业自有代码搜索

支持将企业自有代码库纳入智能代码搜索引擎的搜索范围，通过搜索企业自有的内部代码资源，可以盘活企业代码资产，减少代码的重复编写，避免开发资源浪费。

搜索访问权限控制

支持企业代码分类，根据企业人员的部门、岗位等角色属性设置不同的搜索访问权限，从而实现对企业代码搜索范围的安全管理，保护数据安全。

AIXcoder 大模型系列产品之三



大模型智能知识库

大模型智能知识库

产品功能列表

- 将企业知识内容导入并编辑，构建信息存储，有效管理企业知识资产
- 大模型支撑企业知识内容的训练、搜索、提示工程等核心功能
- 扩展大模型基础能力，提供文档摘要、搜索统计、智能联想、结构排序等关键能力

智能知识库：能力介绍

领域知识问答能力
融入垂直领域的专业语料作为上下文进行问答对话

自然对话能力
以自然语言对话的方式完成同知识库的交互动作

语义检索能力
自动识别用户输入内容的语义信息进行检索

文档摘要能力
基于大模型基础能力提供针对文档的内容摘要功能

语义理解能力
基于注意力机制"读懂" 用户输入的语义信息

内容生成能力
借助大模型融合行业知识，生成领域专业内容

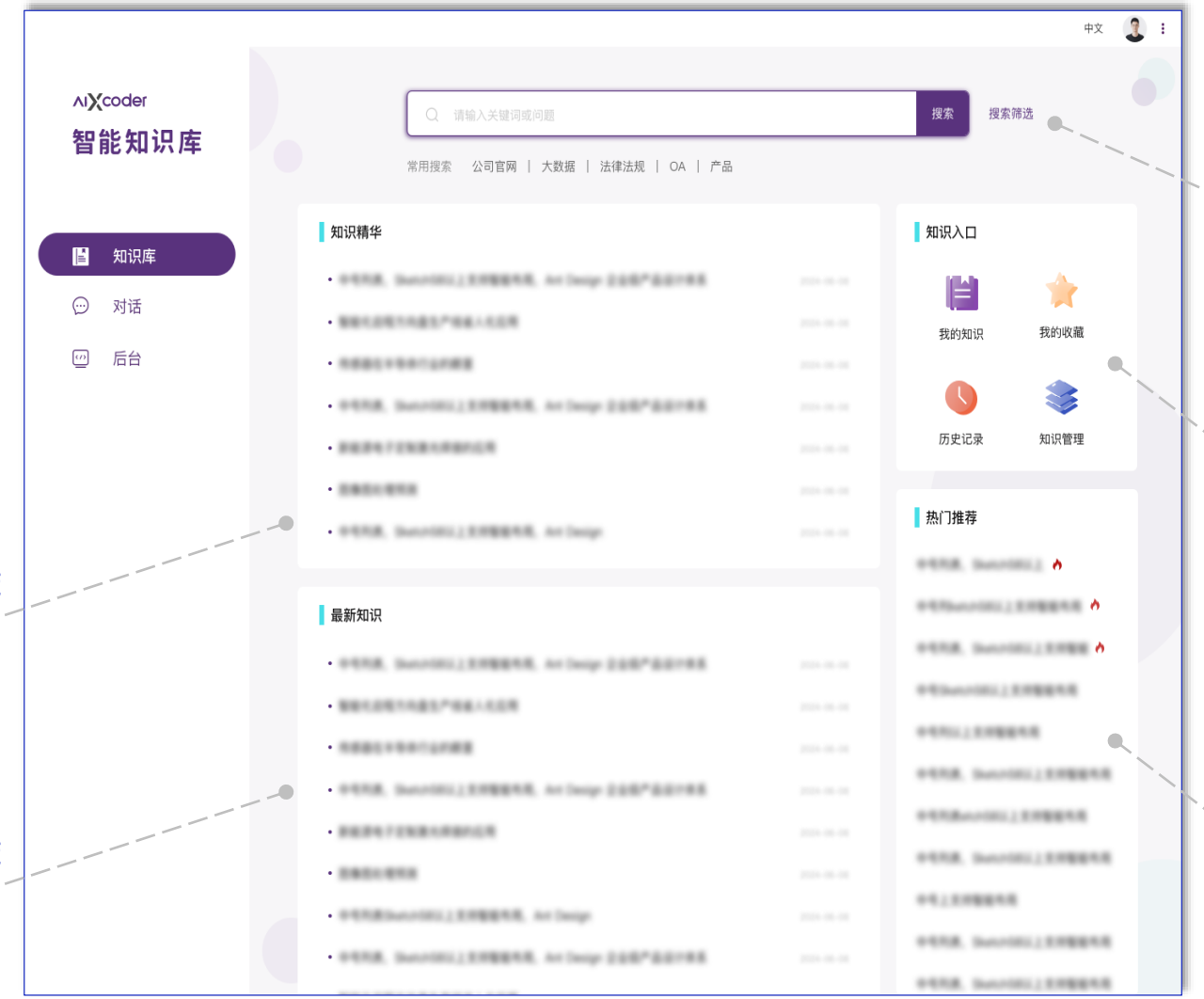
检索优化能力
借助多种优化手段来改善检索效果

知识管理能力
简单易用的知识管理与应用能力

智能知识库 – 知识检索

知识检索

- 提供准确的知识检索与应用能力，支持NLU语义理解和搜索纠错能力
- 以智能识别用户检索意图，提供更加精准的检索结果



支持语义、全文、拼音等多种检索方式，支持搜索纠错

快捷知识入口

精华内容推荐

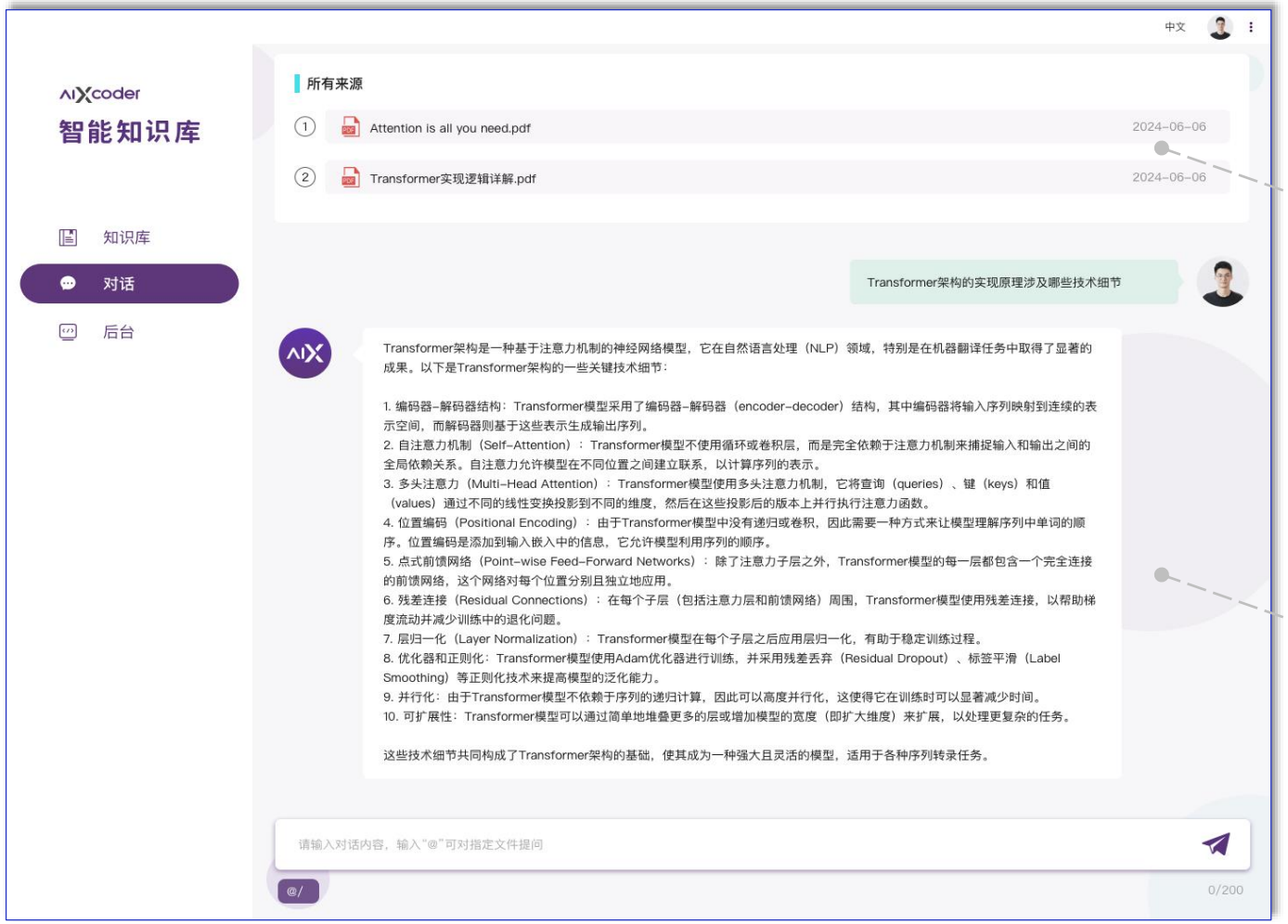
热门知识推荐

最新知识推荐

智能知识库 –智能对话

智能对话

- 以大模型的语义理解与内容生成能力为基础
- 依托于垂直领域内的专业语料素材，提供自然语言对话、领域知识学习和文档问答能力
- 精确生成符合用户要求的内容



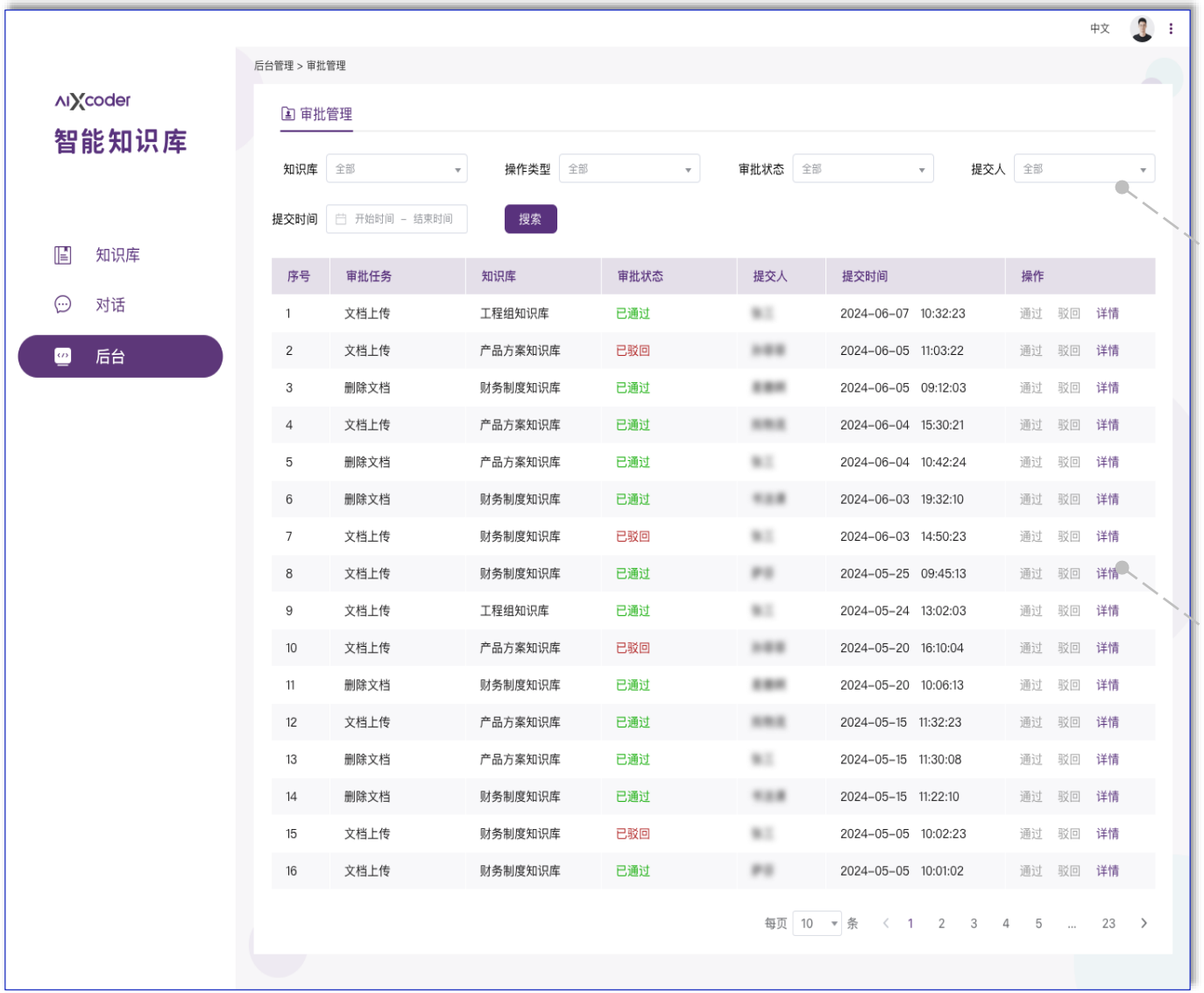
快速阅读理解文档内容，用于后续问答

大模型结合行业内语料，提供专业且符合领域事实要求的指定内容

智能知识库 –后台管理

后台管理

- 提供高效的知识管理能力，支持知识上传、编辑、审批、版本管理功能，方便用户快速管理、更新和维护知识文档。
- 提供多种层级的知识权限管控功能



便捷的知识审批筛选

多维度知识审批列表

优势介绍

精准识别

以大模型基础能力结合领域知识学习能力，精准识别用户意图

专业经验

具备丰富的同行业类似项目落地实施和大模型训练经验

领域沉淀

在人工智能、知识库相关领域的学术研究和产品工程化方面沉淀多年

科研后盾

与北京大学软件研究所共建实验室，具备坚实的科研后盾支撑

AIXcoder

军工大模型落地
需求及挑战

军工大模型落地的需求及挑战

aiXcoder

诉求

- 智能知识库
- 研发模式转型
- 私有化部署
- 小数据量训练
- 降本增效

挑战

- 企业代码数据和代码安全
- 计算资源有限
- 企业领域知识
- 大模型个性化训练

解决方案

- 多应用场景落地
- GPU多样性适配
- 企业数据治理
- 企业管理平台

私有化部署服务

部署方式灵活：

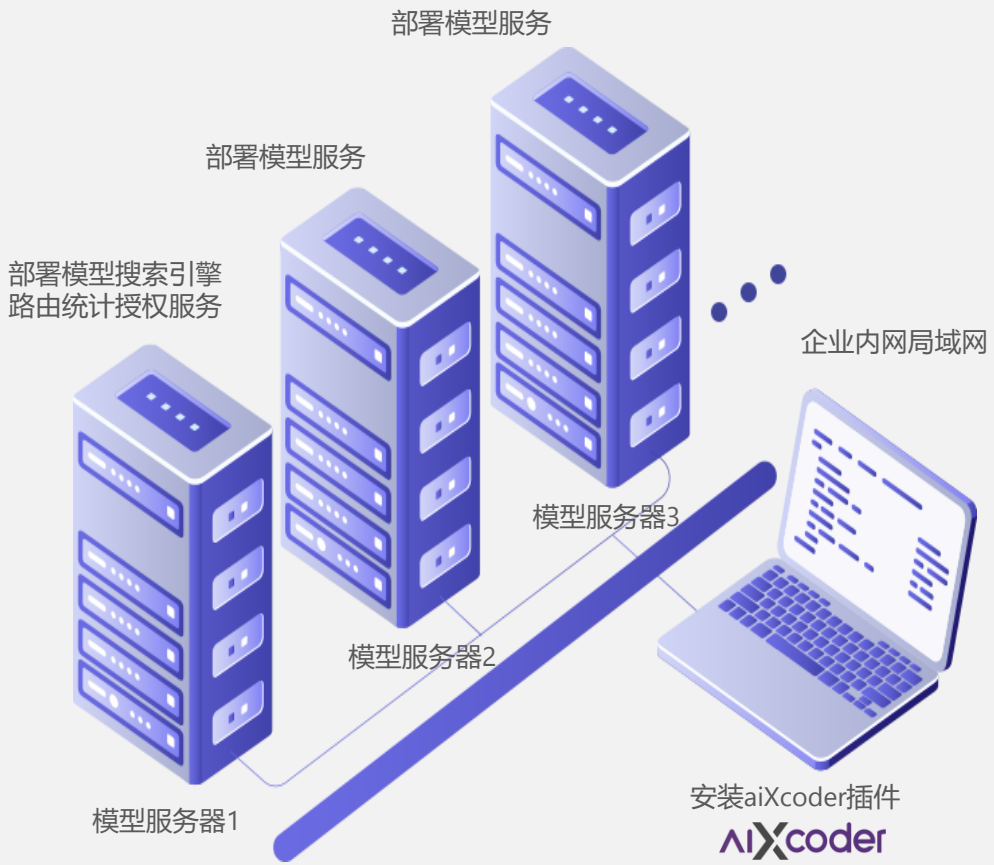
- 支持K8S和Docker部署方式，灵活满足不同客户需求
- 集群式部署，提高系统可用性
- 根据实际用户量，可灵活伸缩服务数量
- 适配国产化操作系统，满足信创要求

计算资源需求低至：

- 1张V100-代码大模型
- 4张A100-知识库大模型

适配多种芯片：

- 英伟达A系列、H系列、V系列GPU
- 华为昇腾
- .....



国产化适配生态

全面支持国产适配

- aiXcoder产品全面支持国产化适配生态，覆盖GPU、CPU、操作系统等软硬件领域
- 通过各个层面的算力优化，大幅提升大模型在训练和推理方面的速度
- 帮助军工企业构建自主安全、高效可落地的大模型环境

HUAWEI
Enflame 燧原科技
摩尔线程 MOORE THREADS
METAX 沐曦
Cambricon 寒武纪
算能 SOPHGO
天数智芯 Iluvatar CoreX
海飞科 HEXAFLAKE
LYNXI
HISILICON
Phytium飞腾
HYGON
Suma
超云 SUPER CLOUD 让数据中心更简单
i(O)百信 TRUST
统信软件技术有限公司 UnionTech Software Technology Co., Ltd.
KYLINSOFT 麒麟软件
……

国产适配

- 华为昇腾
- 百信
- 海光
- 天数智芯

昇腾技术认证书

北京硅心科技有限公司
aiXcoder 智能编程系统
与
华为技术有限公司
Atlas 800T A2 训练服务器
完成并通过相互兼容性测试认证
特授予
HUAWEI COMPATIBLE证书及认证徽标的使用权
有效期：2024年03月－2027年03月

北京硅心科技有限公司
刘洋
联合创始人
签署日期：2024年03月06日

华为技术有限公司
张迪煊
昇腾计算领域总裁
认证机构：昇腾计算产品线
测试机构：杭州智能计算Openlab
授予日期：2024年03月06日
证书编号：E202403013

HUAWEI COMPATIBLE

Ascend
证书编号：BX2024021801
TRUST 百信
aiXcoder

产品兼容性认证
PRODUCT COMPATIBILITY CERTIFICATION

百信信息技术有限公司是国内领先的以安全创新和可信计算技术为主导的基础安全 IT 产品提供商。主要提供基于国产处理器的计算机、服务器及相关软硬件等产品，可广泛应用于党政、金融、能源、交通等关键领域。

北京硅心科技有限公司是一家专注于 AIGC for Code 领域的高新技术企业，致力于将前沿人工智能技术应用于软件工程。通过为企业提供私有化专用代码大模型，提升企业软件开发效率和代码质量，帮助其提升研发效能。

经百信信息技术有限公司测试，得出以下产品兼容性互认结论：aiXcoder 智能编程系统 在百信恒山 AI326TA（CPU 鲲鹏 920 NPU 昇腾 910B）上经过测试，双方产品互相兼容，运行稳定，满足用户需求。特此认证。

百信信息技术有限公司
2024 年 2 月 18 日

北京硅心科技有限公司
2024 年 2 月 18 日

aiXcoder Suma

产品兼容性互认证

北京硅心科技有限公司（以下简称“硅心科技”）自主研发的aiXcoder智能编程系统是主要面向金融、电信、能源、交通、政府、教育、制造等行业打造的智能化软件研发产品，用于软件开发、软件测试、软件需求、软件设计、系统运维等应用场景，为企业提供“AI+软件工程”的智能化服务系统。

中科可控信息产业有限公司（以下简称“中科可控”）基于海光处理器研发的X系列异构加速服务器，搭载海光DCU加速卡，围绕深度学习算法提供异构加速性能，以灵活的架构、丰富的扩展，满足人工智能GPU集群的各种应用需求。

经**硅心科技**与**中科可控**联合测试，**aiXcoder智能编程系统** 与 **X73系列、X74系列、X78系列异构加速服务器** 兼容性良好，可以顺利安装、配置，并稳定运行。

北京硅心科技有限公司

中科可控信息产业有限公司

证书编号：PCC-SUMA-S08-4352-01
认证日期：2024 年 04 月 17 日

天数智芯 Iluvatar CoreX
aiXcoder

产品兼容性认证书

经过上海天数智芯半导体有限公司和北京硅心科技有限公司的联合测试，得出以下结论：

上海天数智芯半导体有限公司的智铠100加速卡（**MR-V100**）与**aiXcoder智能编程系统**完成兼容性测试，能够达到兼容性要求，整体运行稳定，支持良好，可以满足用户的应用需求。

特此证明！

上海天数智芯半导体有限公司

北京硅心科技有限公司

证书编号：PCC-ILUVATAR-20240403-001
认证日期：2024 年 4 月 3 日

为什么军工企业需要个性化训练 - 必要性

为什么军工领域需要个性化训练?

- 具有很强的业务相关逻辑
- 拥有明确的编码规范
- 拥有独特的代码风格
- 拥有自定义专有名词、缩写、API调用逻辑等等

打榜题 vs 真实业务开发

```python
def incr_list(l: list):
    """Return list with elements incremented by 1.
    >>> incr_list([1, 2, 3])
    [2, 3, 4]
    >>> incr_list([5, 3, 5, 2, 3, 3, 9, 0, 123])
    [6, 4, 6, 3, 4, 4, 10, 1, 124]
    """
    return [i + 1 for i in l]
```

```python
def solution(lst):
    """Given a non-empty list of integers, return the sum of all of the odd elements
    that are in even positions.

    Examples
    solution([5, 8, 7, 1]) ==>12
    solution([3, 3, 3, 3, 3]) ==>9
    solution([30, 13, 24, 321]) ==>0
    """
    return sum(lst[i] for i in range(0,len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```

```python
def encode_cyclic(s: str):
    """
    returns encoded string by cycling groups of three characters.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group. Unless group has fewer elements than 3.
    groups = [(group[1:] + group[0]) if len(group) == 3 else group for group in groups]
    return "".join(groups)

def decode_cyclic(s: str):
    """
    takes as input string encoded with encode_cyclic function. Returns decoded string.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group.
    groups = [(group[-1] + group[:-1]) if len(group) == 3 else group for group in groups]
    return "".join(groups)
```

```cpp
};

// 判断并设置战斗机的系统配置
void configureSystem(CombatEnvironment env) {
    // 若环境类型为电子战且使用对外通讯频道
    if (env.environmentType == "电子战" && env.isPublicCommunication) {
        // 则归属系统为EW（电子作战系统）
        std::cout << "归属系统为EW（电子作战系统）" << std::endl;
    }
    // 若环境类型为侦察或对地攻击
    else if (env.environmentType == "侦察" || env.environmentType == "对地攻击") {
        // 则归属系统为CS（作战支持系统）
        std::cout << "归属系统为CS（作战支持系统）" << std::endl;
    }
    // 其他情况
    else {
        // 归属系统为NS（导航系统）
        std::cout << "归属系统为NS（导航系统）" << std::endl;
    }
}

int main() {
    // 创建一个作战环境实例，设置环境类型为"电子战"，并使用对外通讯
    CombatEnvironment env = {"电子战", true};

    // 调用函数配置系统
    configureSystem(env);
```

企业个性化训练

企业领域知识

+

aiXcoder基座代码大模型
Code LLM

进行个性化训练

- ✔ 主干模型的推理能力不受影响
- ✔ 较小的微调参数
- ✔ 训练成本少，训练成本可控
- ✔ 在个性化数据上获得更好的效果

企业专属代码大模型

- 生成代码符合企业业务逻辑
- 生成代码符合企业编码规范和风格
- 生成代码高可用率

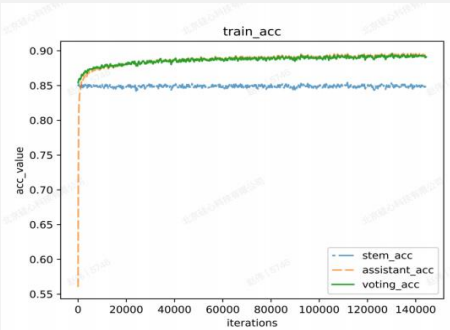
个性化训练 – 企业专属测评集

评估指标： Exact Match　BLEU-4　CODE BLEU　Acceptable

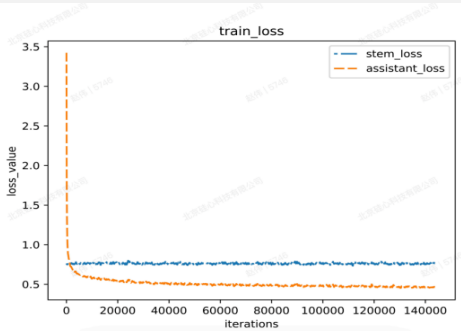
评估流程：

训练集与验证集的划分 → 评测集构建-上下文与标准答案 → 模型调用-输出结果记录 → 多种评测指标打分-统计结果

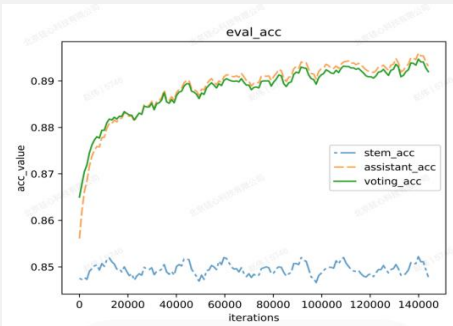
企业领域集上的准确率



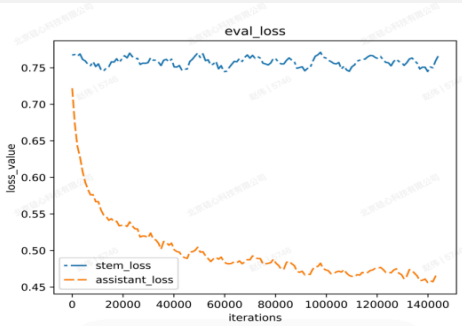
企业领域集上的损失



企业领域验证集上的准确率



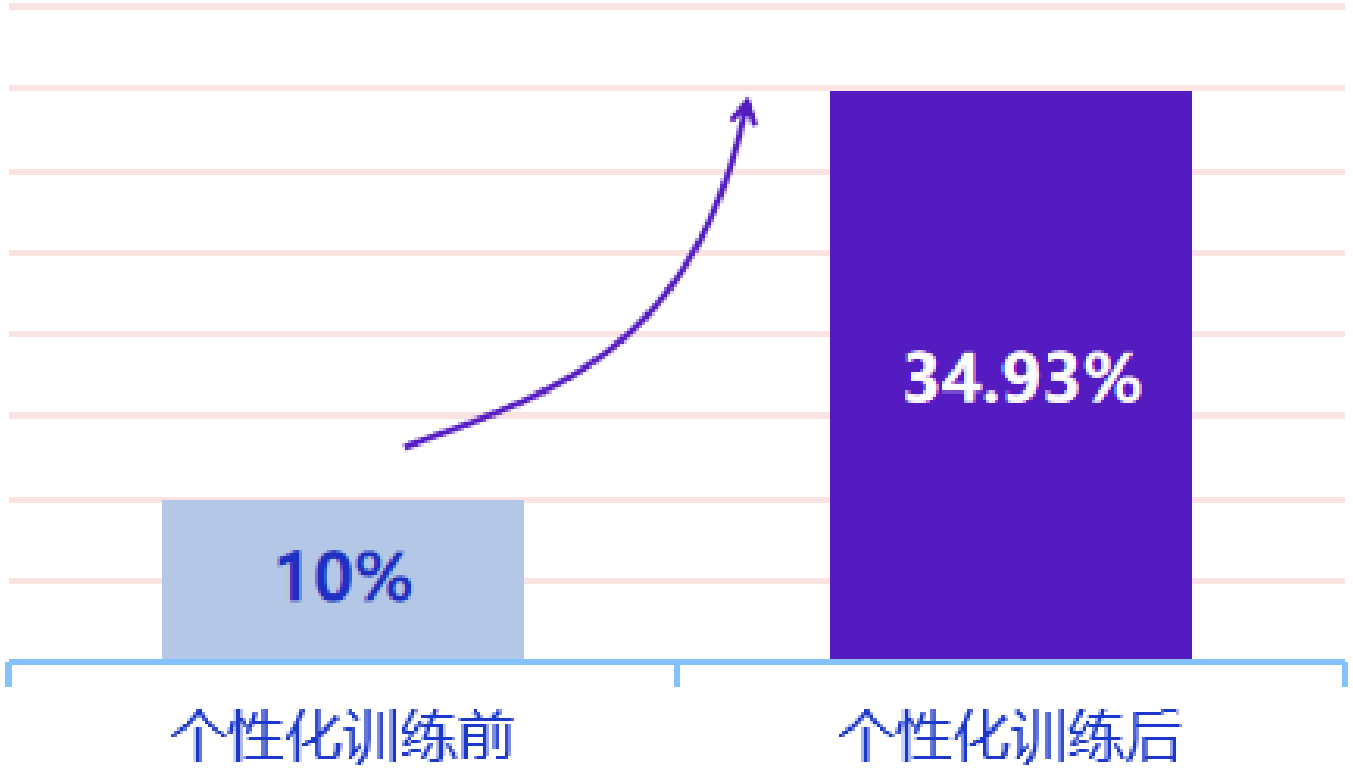
企业领域验证集上的损失



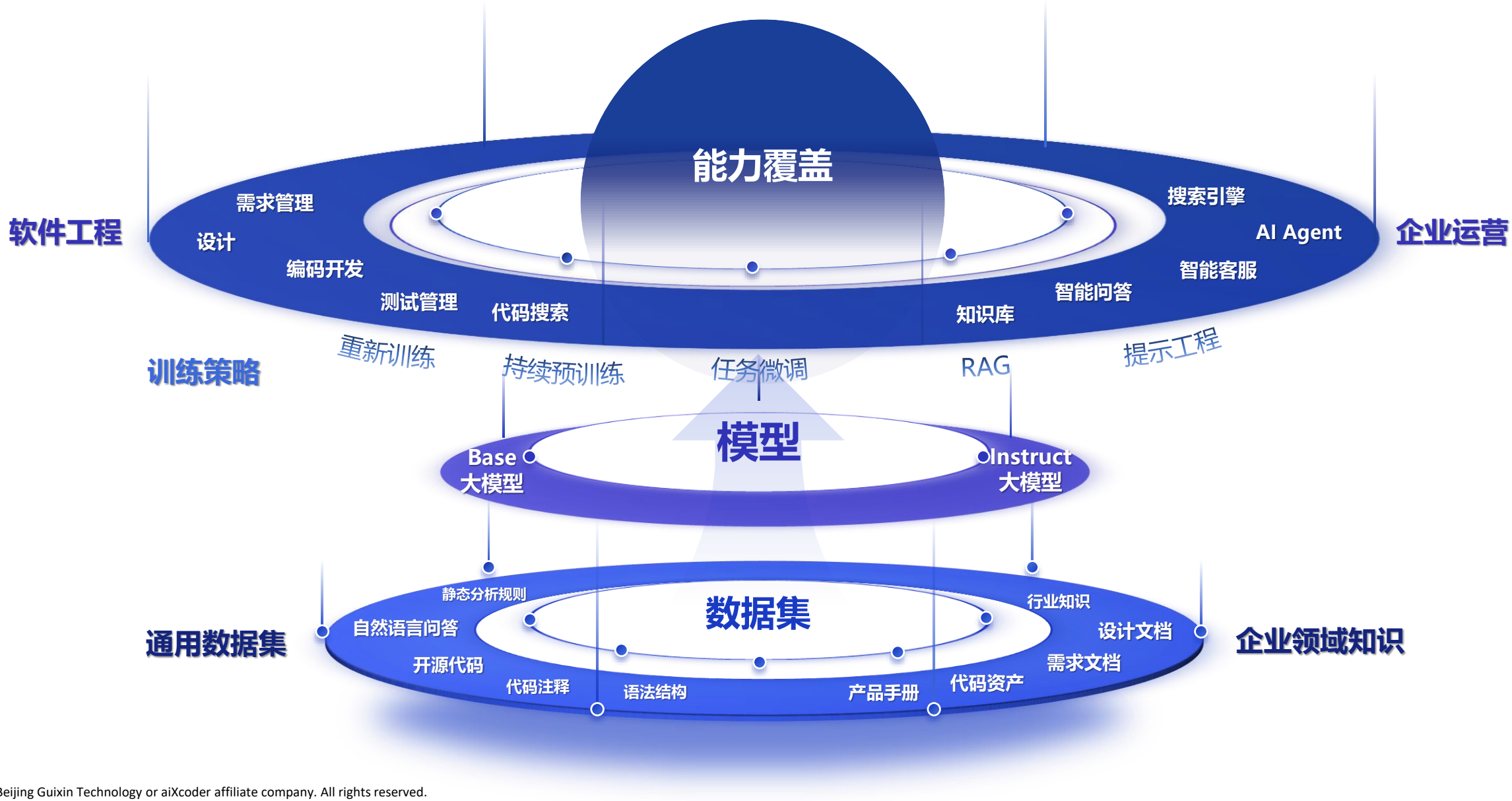
避免常规“打榜”评测集：不使用如HumanEval、MBPP、MultiPL-E等常规的“打榜”评测集

小数据量个性化训练真实成果

仅代码训练得到的大幅提升



大模型落地 - 基于领域知识的大模型构建



AIXcoder

大模型实践案例

应用案例：航天某所-软件开发和知识检索

智能开发与知识检索平台

将大模型通用能力与航天控制软件领域知识相结合，为航天某院提供软件智能化开发与知识检索能力

1. **领域代码特点分析**：全部使用C语言，遵守自有编程规范，大量航天控制相关的术语与软件模块。
2. **专属代码模型训练**：根据领域代码特点，在所内训练了专属的智能化代码补全模型。
3. **智能知识检索工具**：充分利用已有的各类数据资产，包括领域知识文档、软件代码资产、测试用例文档等，通过大模型从数据资产中挖掘启示价值，构建大模型知识库检索/问答系统和代码生成系统。
4. **所内私有化部署**：根据所内服务器与网络环境，部署与适配系统。

- 通过将辅助编程和知识检索能力与航天控制领域知识相结合，用户可以更加**高效地完成编程任务，提升代码质量；更精准地完成领域知识查询，实现知识积累、共享和价值应用。**

应用案例：航天某院-数据公式与伪代码自动生成

	符号	维度	数据类型	对应软件变量
输入量	t_{GPS}	1*1	float64	sGpsData.tGps
	dTj	1*1	float64	sGpsData.dTj
	mtdTj_J2000	1*1	float64	sOrbitInjectPara.tdTjJ2000
	mtj2000UpLimit	1*1	float64	sGpsOrbitEps.mtJ2000UpLmt
	mtj2000LowLimit	1*1	float64	sGpsOrbitEps.mtJ2000DownLmt
输出量	tJ2000	1*1	float64	sGpsData.tJ2000
	C_{PN}	3*3	float64	sGpsData.Cpn[3][3]
	T_{CD}	3*3	float64	sGpsData.Tcd[3][3]
局部变量	tmp1	1*1	float64	
	tmp2	1*1	float64	
常量				
底层库函数	CalcCPN	3*3	float64	void CalcCpn(float64 *cpn, float64 tj2000)
	{C_{PN}}^{T}	3*3	float64	void MatrixTran33(float64 *matrixT, float64 *matrixRaw)

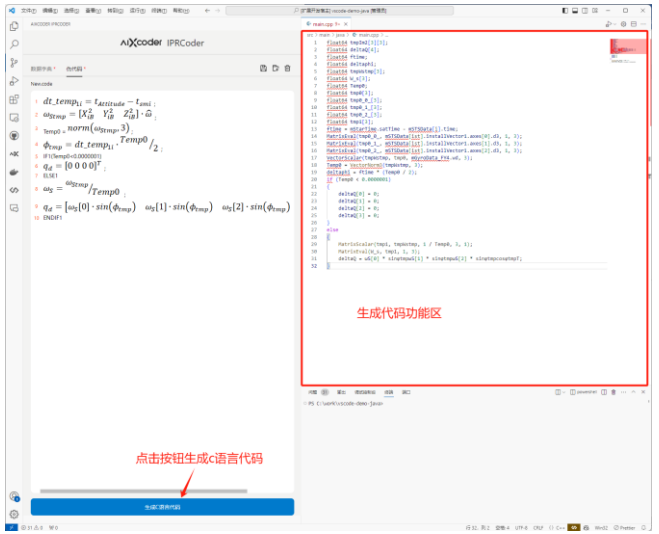
伪代码

IF1(RVDSenFlag_G=AAh) //地面指定敏感器
{ Flg_3chose2=55h;
IF2(RVDS5km_G=11h 且 Flg_RGPSfilter=1)
{RVDSensor=RGPS; Cnt_ δ max5km=0; }
ELSEIF2(RVDS5km_G=22h 且 Flg_WRfilter=1)
{RVDSensor=WR; Cnt_ δ max5km=0; }
ELSEIF2(RVDS5km_G=33h 且 Flg_LRfilter=1)
{RVDSensor=LR; Cnt_ δ max5km=0; }
ELSE2 {RVDSensor=NoSensor}
ENDIF2
}
ELSE1
{RVDSensor=NoSensor}
ENDIF1

IF1(RVDSensor=NoSensor)
{//三取二保证用于用于导航的敏感器数据质量高
Flg_3chose2=55h;
IF2（Flg_RGPSfilter=1 且 Flg_WRfilter=1 且 Flg_LRfilter=1 且 RVDSensorSelect5km_G=AAh)
{ Flg_3chose2=AAh;

$$\begin{matrix}\hat{x}_{WRr}-\hat{x}_{Rr}\\ \delta\,1temp=[\hat{y}_{WRr}-\hat{y}_{Rr}];\quad \delta\,1=norm(\delta\,1temp,3)\\ \hat{z}_{WRr}-\hat{z}_{Rr}\end{matrix}$$

$$\begin{matrix}\hat{x}_{WRr}-\hat{x}_{Gr}\\ \delta\,2temp=[\hat{y}_{WRr}-\hat{y}_{Gr}];\quad \delta\,2=norm(\delta\,2temp,3)\\ \hat{z}_{WRr}-\hat{z}_{Gr}\\ \hat{x}_{Gr}-\hat{x}_{Rr}\end{matrix}$$




```
float64 gammaVect3[3];
unint32 pSTSUseable;
float64 tmp0[3];
float64 tmp1[3];
float64 tmp2[3];
float64 tmp3;
float64 tmp4[3];
float64 tmp5[3];
float64 tmp6[3];
float64 tmp7;
float64 tmp8[3];
float64 tmp9[3];
MatrixScalar(tmp0, morbit.vorbit, 1/299792.458, 3, 1);
MatrixEval(gammaVect3, tmp0, 1, 3);
if (pSTSUseable == 1) {
    MatrixAdd(tmp1, mSTSData[i].lmeasure.axes[2].d3, gammaVect3);
    MatrixTran(tmp2, mSTSData[i].lmeasure.axes[2].d3, 3, 1);
    MatrixMulti(tmp3, tmp2, gammaVect3, 1, 3, 1);
    MatrixMulti(tmp4, mSTSData[i].lmeasure.axes[2].d3, tmp3, 3, 1, 1);
    MatrixSub(mSTSData[i].lmeasure.axes[2].d3, tmp1, tmp4, 3, 1);
    MatrixScalar(tmp5, mSTSData[i].lmeasure.axes[2].d3, 1/VectorNormN(mSTSData[i].lmeasure.axes[2].d3, 3), 3, 1);
    MatrixEval(mSTSData[i].lmeasure.axes[2].d3, tmp5, 1, 3);
    MatrixTran(tmp6, mSTSData[i].lmeasure.axes[0].d3, 3, 1);
    MatrixMulti(tmp7, tmp6, gammaVect3, 1, 3, 1);
    MatrixMulti(tmp8, mSTSData[i].lmeasure.axes[2].d3, tmp7, 3, 1, 1);
    MatrixSub(mSTSData[i].lmeasure.axes[0].d3, mSTSData[i].lmeasure.axes[0].d3, tmp8, 3, 1);
    MatrixScalar(tmp9, mSTSData[i].lmeasure.axes[0].d3, 1/VectorNormN(mSTSData[i].lmeasure.axes[0].d3, 3), 3, 1);
    MatrixEval(mSTSData[i].lmeasure.axes[0].d3, tmp9, 1, 3);
}
```

C语言代码

应用案例：某核能研究院

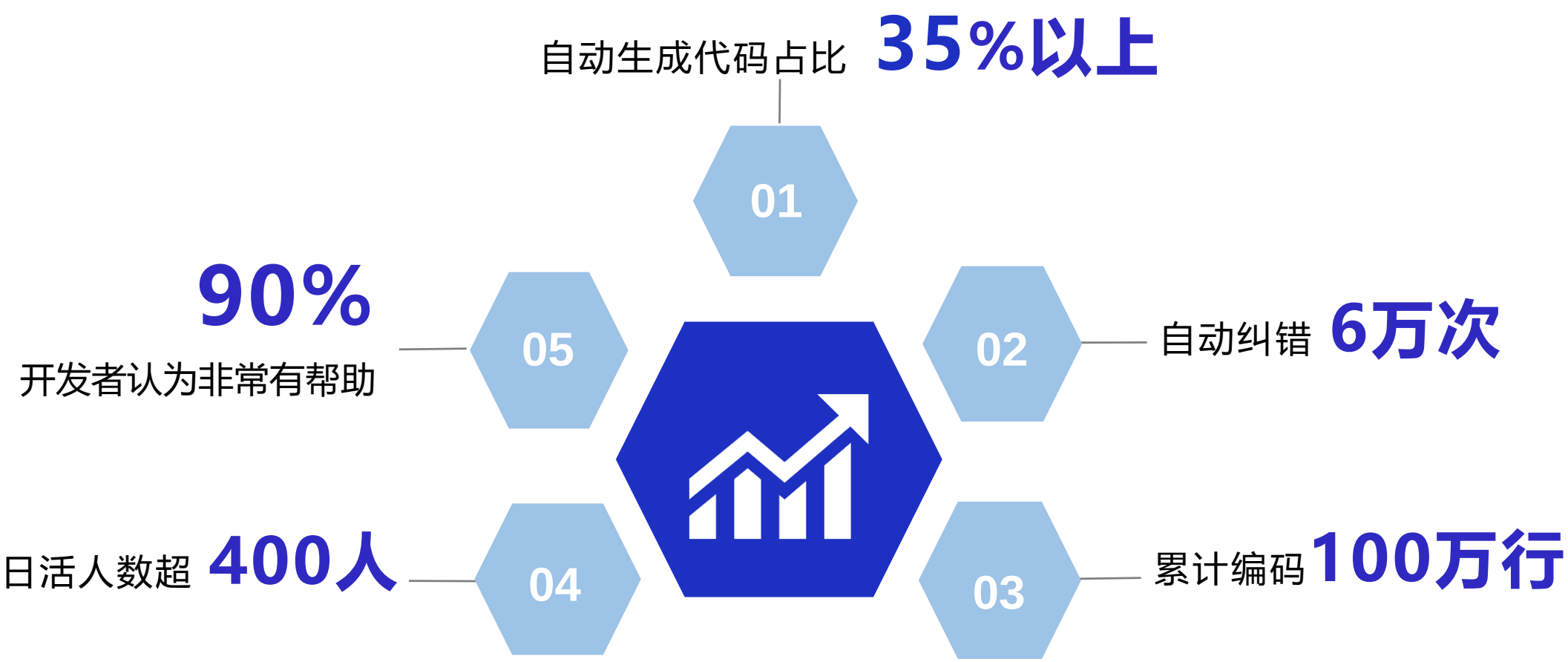
智能代码构造与分享平台

将智能代码搜索引擎与定制化搜索需求结合，为某核能研究院开发“智能代码构造与分析平台”

1. **领域需求分析：**根据某院需求，代码搜索引擎添加代码缺陷样例库，支持对缺陷的搜索与API获取。
2. **构建专属搜索引擎：**根据特定需求，定义缺陷提取规范，构建专属搜索引擎。
3. **开发智能化软件开发模块：**提供IntelliJ IDEA客户端，结合搜索引擎定制需求，实现客户端发起搜索功能。实现了智能化代码搜索与院内已有系统的对接。
4. **系统私有化部署：**根据服务器与网络环境，制定部署方案，实施系统安装部署。

通过将辅助编程功能与航空行业特点相结合，开发人员可以更加高效地完成编程任务，降低出错率，提升代码质量，从而为航空行业的发展贡献力量。同时，这也有助于提高航空行业的整体竞争力，推动行业的数字化转型进程。

应用案例：某企业大规模应用



某大型企业研发中心 3000+人 使用

当前和多家军工院所合作的其他场景

- ✓ 军工企业智能知识库系统
- ✓ 基于大模型的智能化测试平台
- ✓ 作战模拟与仿真
- ✓ 决策链信息辅助支持系统
- ✓ 大模型驱动的应急系统APP自动生成
- ✓ 大模型驱动的情报智库
- ✓ ……

AIXcoder

赋能军工企业，落地领域大模型

助力军工行业，开创智能化未来

扫码联系 AIXcoder