CodeGeeX

清華大學 Tsinghua University

CodeGeeX: 自动代码生成插件

主讲人：郑勤锴

体验DEMO

GitHub仓库

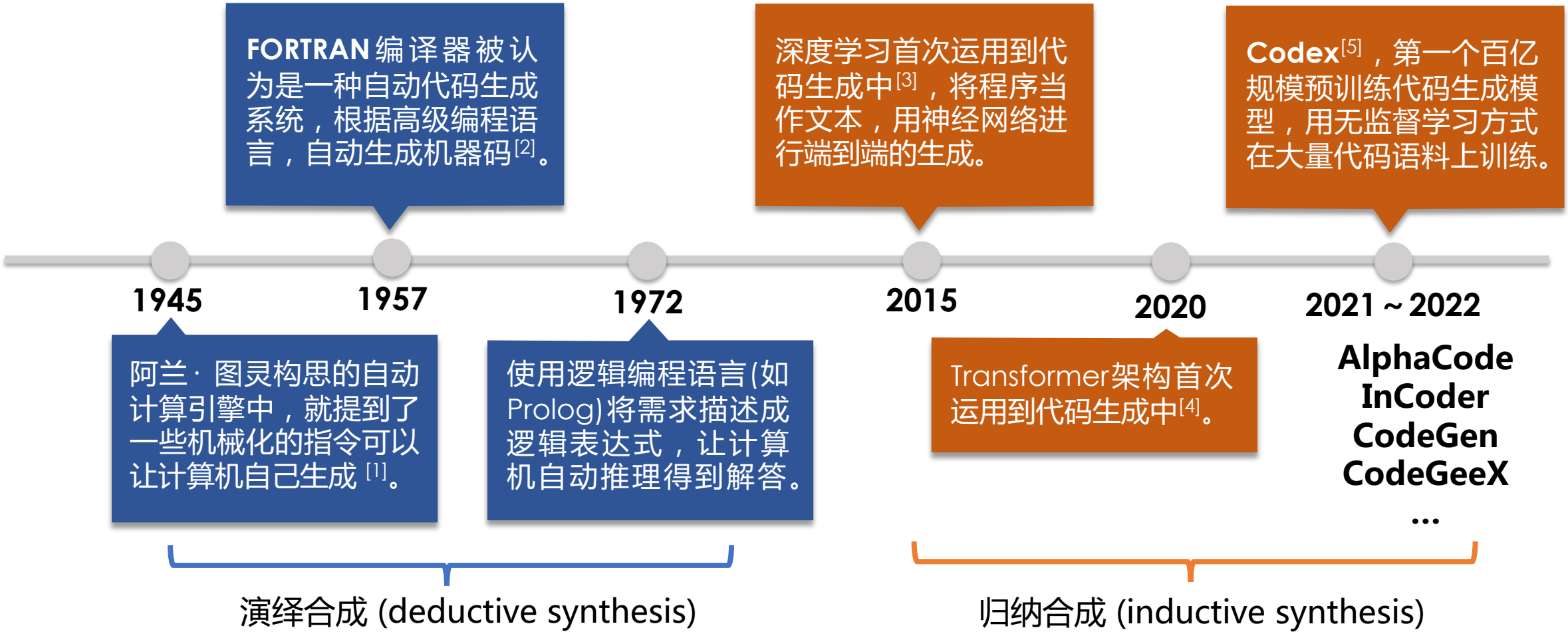
主页: https://models.aminer.cn/codegeex

源码: https://github.com/THUDM/CodeGeeX

插件: VS Code插件市场搜索“codegeex”免费下载

背景：自动代码生成

自动代码生成，或程序合成 (program synthesis)，是计算机科学领域长久以来的一大难题：



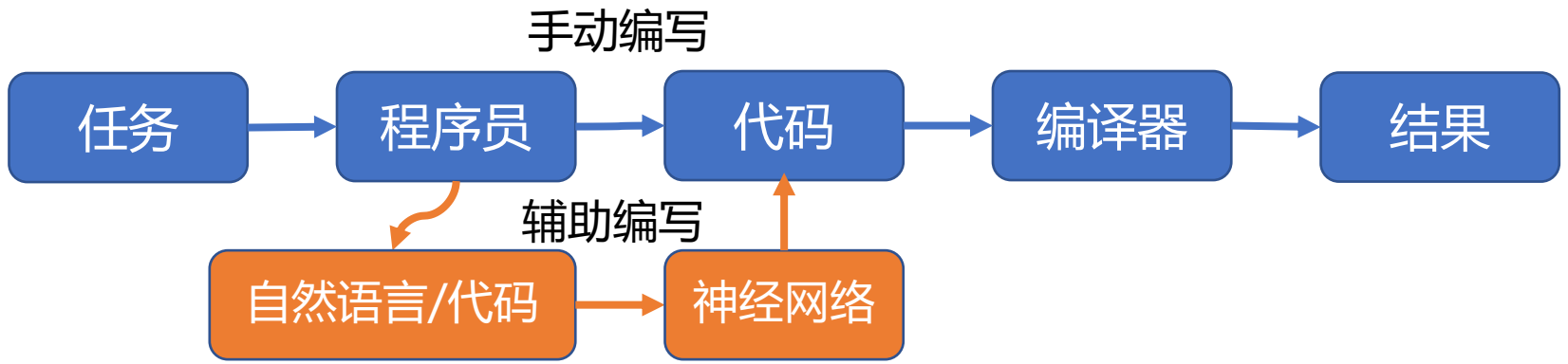
自动代码生成进入新阶段：预训练模型使生成复杂的、正确的代码成为可能。

[1] B.J. Copeland, Alan Turing's Electronic Brain: The Struggle to Build the ACE, the World's Fastest Computer, 2012
[2] Introduction to Program Synthesis Fall 2022, https://people.csail.mit.edu/asolar/SynthesisCourse/index.htm
[3] Mou L, Men R, Li G, et al. On end-to-end program generation from user intention by deep neural networks[J]. arXiv preprint arXiv:1510.07211, 2015.
[4] Svyatkovskiy A, Deng S K, Fu S, et al. Intellicode compose: Code generation using transformer. 2020
[5] Chen M, Tworek J, Jun H, et al. Evaluating large language models trained on code[J]. arXiv preprint arXiv:2107.03374, 2021.

背景：自动代码生成

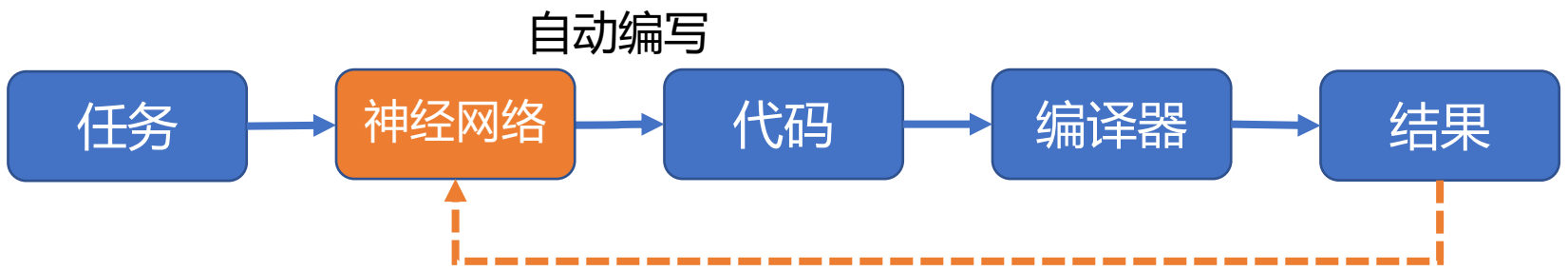
1. 一般编程过程

2. AI辅助编程过程

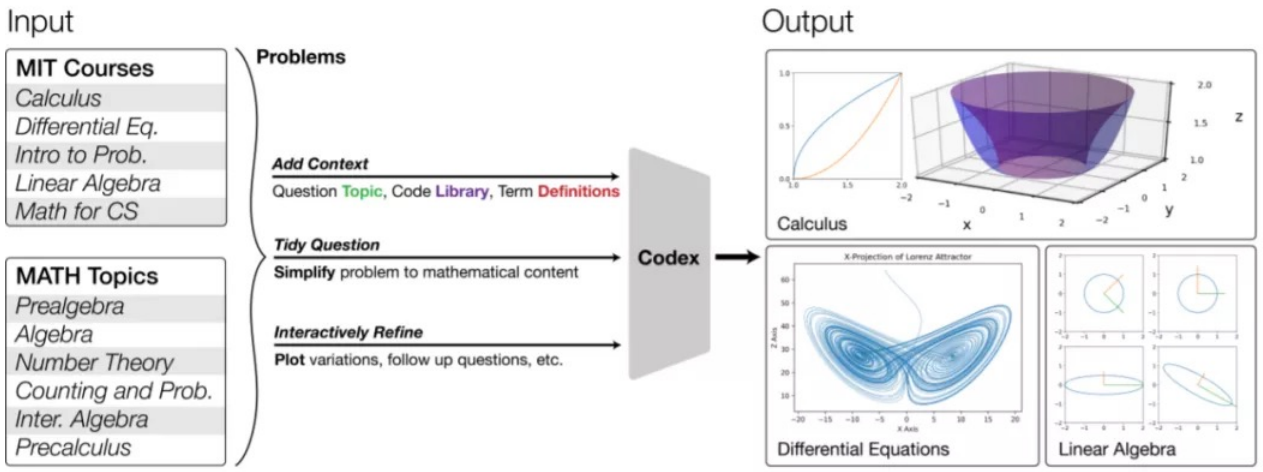


CodeGeeX

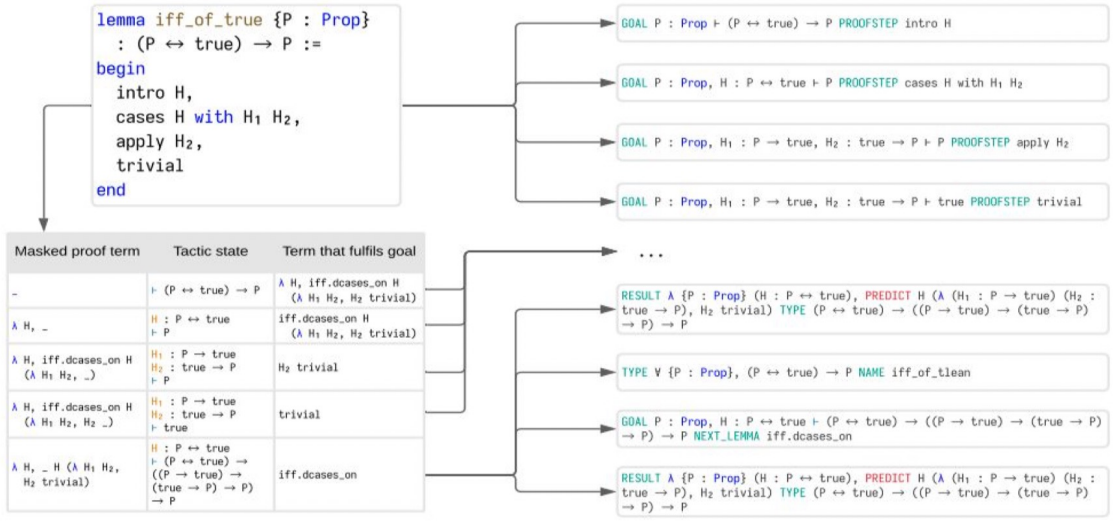
3. AI自动编程过程



背景：代码生成模型的应用

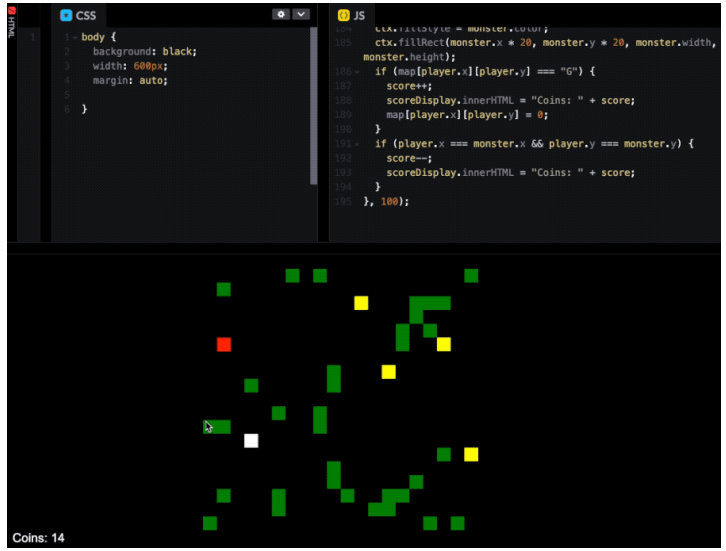


使用Codex解决MIT本科数学题[2]



形式化数学定理证明[3]

- **复杂问题求解**
  - AlphaCode解决竞赛级编程题[1]
  - Codex解决MIT本科数学题[2]
- **数学定理证明**
  - PACT[3]
- **前端代码生成**
- **游戏生成**
- **文档生成**
- **程序修复**
- …

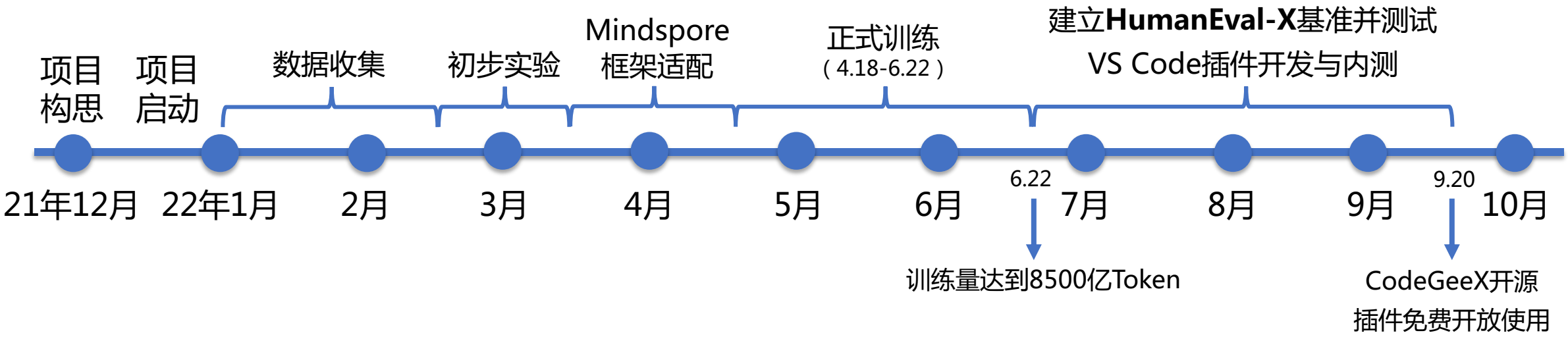


Codex生成的“极简塞尔达”游戏

[1] Li Y, Choi D, Chung J, et al. Competition-level code generation with alphacode[J]. arXiv preprint arXiv:2203.07814, 2022.
[2] Drori I, Zhang S, Shuttleworth R, et al. A neural network solves, explains, and generates university math problems by program synthesis and few-shot learning at human level[J]. Proceedings of the National Academy of Sciences, 2022, 119(32): e2123433119.
[3] Han J M, Rute J, Wu Y, et al. Proof artifact co-training for theorem proving with language models[J]. arXiv preprint arXiv:2102.06203, 2021.

CodeGeeX: 开源的大规模多语言代码生成模型

- 由**清华大学知识工程实验室**研发，鹏城实验室提供算力支持，智谱AI、华为MindSpore提供技术支持。
- CodeGeeX是一个具有**130亿**参数的多编程语言代码生成预训练模型，采用华为MindSpore框架实现，使用鹏城实验室“鹏城云脑II”平台中**192节点**昇腾910 AI处理器，在**20多种**编程语言的代码语料库历时**两个月**训练而成。开源开放，支持昇腾和英伟达平台，具有高精度代码生成、代码翻译等能力。



上线以来：CodeGeeX插件服务了**3500+**用户，累计调用量超过**300万次**；

后续计划：优化插件体验，支持多种平台，建立代码生成领域第一的开源社区；

长期目标：将CodeGeeX打造成最好的开源代码生成工具，提高广大程序员的开发效率！

CodeGeeX: 开源的大规模多语言代码生成模型

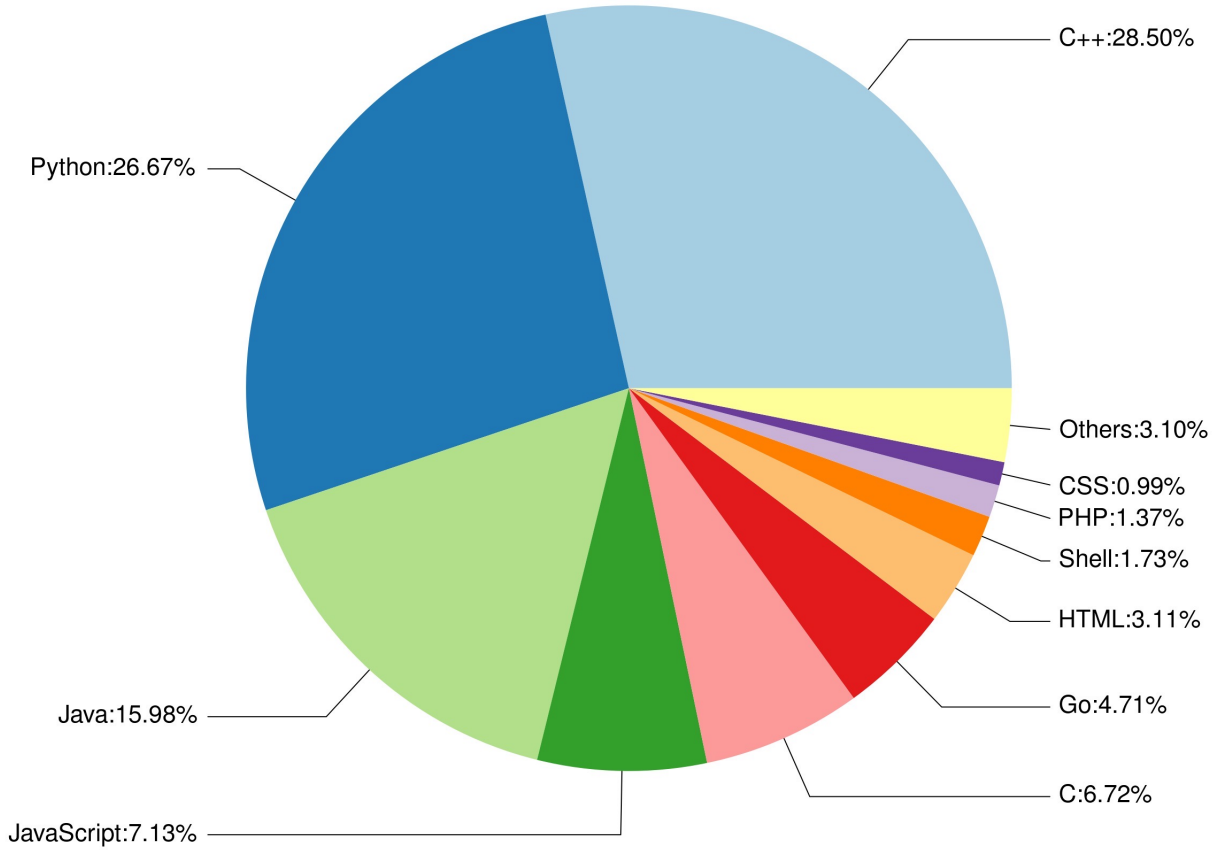
- 大规模代码数据收集
- CodeGeeX模型架构
- CodeGeeX模型训练及优化
- CodeGeeX模型评估
- CodeGeeX自动编程插件
- CodeGeeX开源计划

CodeGeeX: 开源的大规模多语言代码生成模型

- **大规模代码数据收集**
  - 开源数据集+额外爬取数据
  - 总计**23种编程语言**， 涵盖Python, Java, C++, JavaScript, C, Go, HTML等主流语言
- **CodeGeeX模型框架**
- **CodeGeeX模型训练及优化**
- **CodeGeeX模型评估**
- **CodeGeeX自动编程插件**
- **CodeGeeX开源计划**

大规模代码数据

- 训练数据分为**两个部分**：
  - 开源数据集
    - The Pile (代码子集，多语言)[1]
    - CodeParrot (Python)[2]
  - 额外爬取数据集
    - GitHub上带star的开源仓库
    - 按照规则进行筛选和清洗
- 总计**23种编程语言**，涵盖Python, Java, C++, JavaScript, C, Go, HTML等主流语言；
- 标识符转化 (Tokenization)
  - 使用和GPT相同的BPE tokenizer[3]；
  - 词表大小：51200；
  - 数据总量：1587亿tokens；



训练数据各语言token数占比

[1] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. *The pile: An 800gb dataset of diverse text for language modeling.* arXiv preprint arXiv:2101.00027, 2020.
[2] CodeParrot https://github.com/huggingface/transformers/tree/main/examples/research_projects/codeparrot
[3] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374, 2021.

大规模代码数据处理

- 为不同语言的文件加上语言标签

```
// language: C++
// language: C
// language: C#
// language: Cuda
// language: Objective-C
// language: Objective-C++
# language: Python
// language: Java
// language: Scala
% language: TeX
<!--language: HTML-->
// language: PHP
// language: JavaScript
// language: JavaScript
// language: TypeScript
// language: Go
# language: Shell
// language: Rust
/* language: CSS */
-- language: SQL
// language: Kotlin
// language: Pascal
# language: R
!language: Fortran
-- language: Lean
```

- 将代码数据分词并标识符化（Tokenization）

```
// language: Go
// Return list of all prefixes from shortest to longest of the input string
// >>> AllPrefixes('abc')
// ['a', 'ab', 'abc']
func AllPrefixes(str string) []string{
    result := []string{}
    for i = 1; i <= len(str); i ++ {
        result = append(result, str[0: i])
    }
    return result
}
```

```
tensor([[  198,  1003,  3303,    25,  1514,   198,  1003,  8229,  1351,   286,
           477, 21231,   274,   422, 35581,   284, 14069,   286,   262,  5128,
          4731,   198,  1003, 13163,  1439, 36698,   844,   274, 10786, 39305,
         11537,   198,  1003, 37250,    64,  3256,   705,   397,  3256,   705,
         39305, 20520,   198, 20786,  1439, 36698,   844,   274,     7,  2536,
          4731,     8, 17635,  8841,    90,   628, 50268, 20274, 19039, 17635,
          8841,    90,    92,   198, 50268,  1640,  1312,   796,   352,    26,
          1312, 19841, 18896,     7,  2536,  1776,  1312, 19969,  1391,   198,
         50272, 20274,   796, 24443,     7, 20274,    11,   965,    58,    15,
            25,  1312, 12962,   198, 50268,    92,   198, 50268,  7783,  1255,
           198,    92,   198]])
```

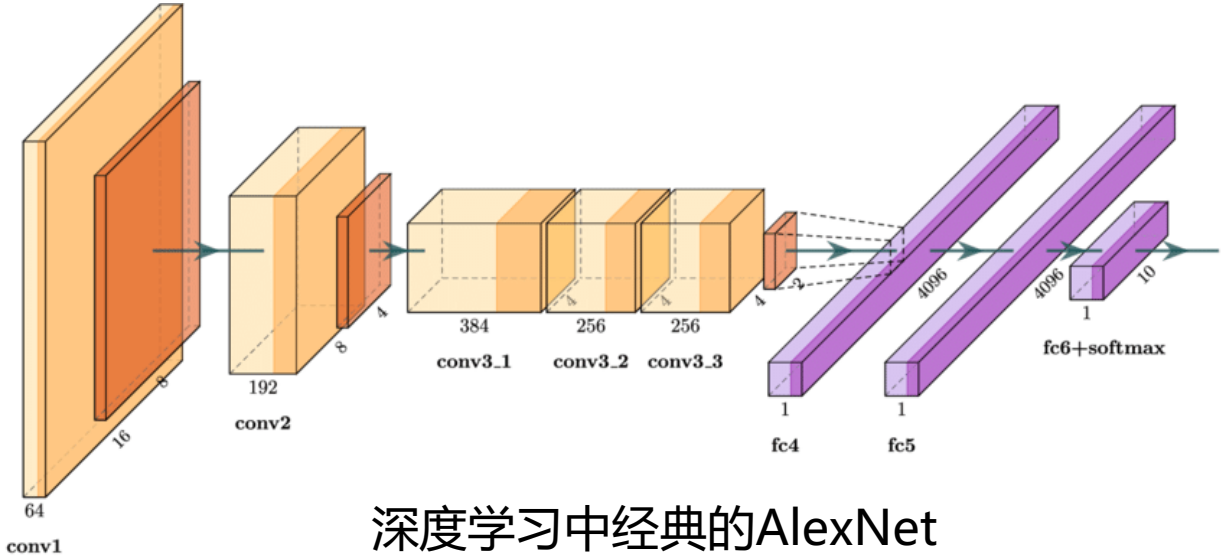
CodeGeeX: 开源的大规模多语言代码生成模型

- 大规模代码数据收集
- **CodeGeeX模型架构**
  - 基于transformer的自回归生成式模型，总计130亿参数
  - 使用自然语言或代码token作为输入，输出下一个token的概率
- CodeGeeX模型训练及优化
- CodeGeeX模型评估
- CodeGeeX自动编程插件
- CodeGeeX开源计划

传统机器学习 vs. 大规模预训练模型

- **传统的机器学习/深度学习流程**

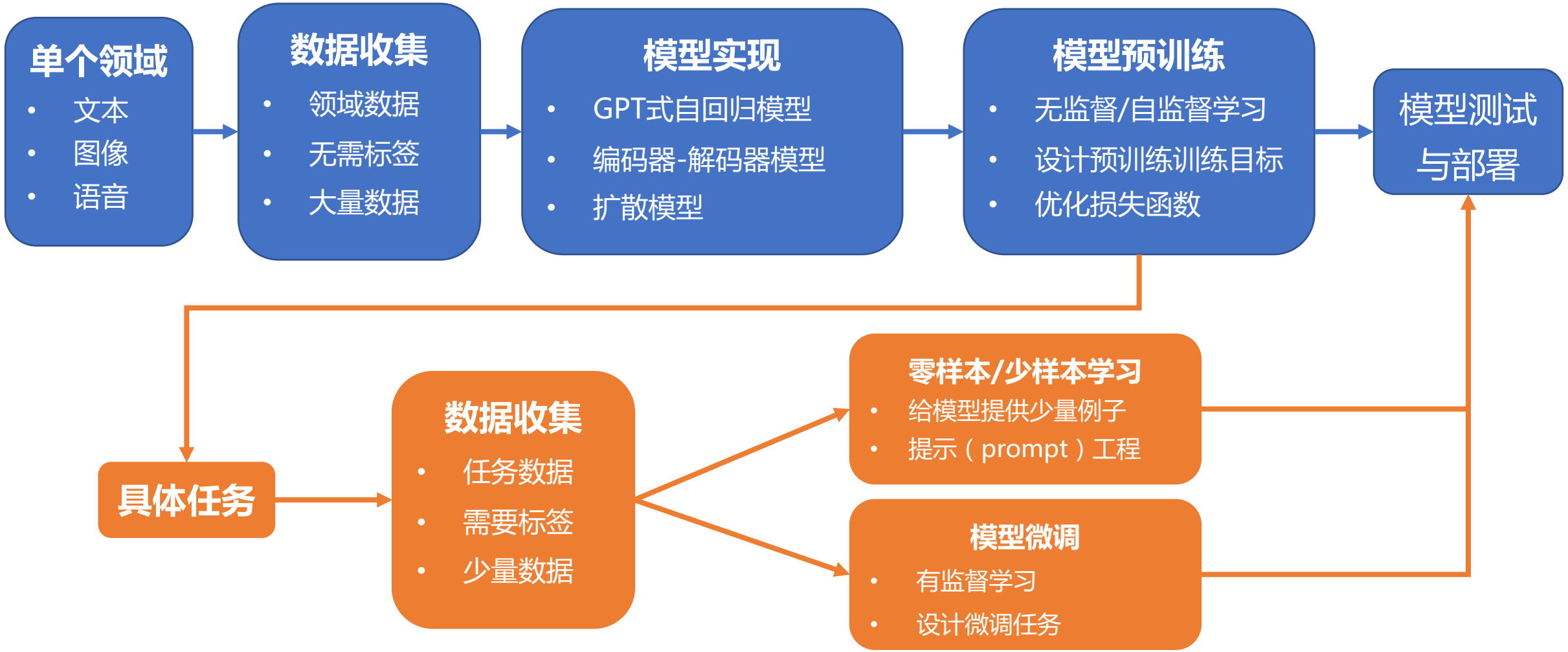
- **存在问题**
  - 对标签数据要求高；
  - 模型难以迁移到其他场景；
  - 一般用于分类/回归任务；
  - 难以利用到更广泛的知识；



深度学习中经典的AlexNet

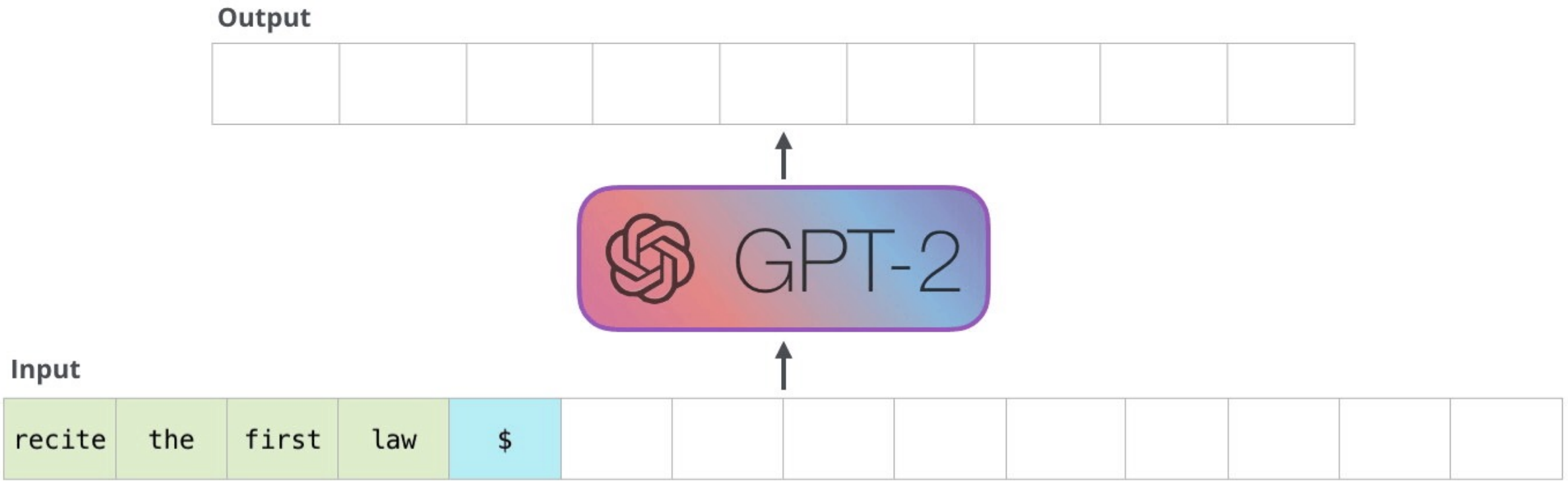
传统机器学习 vs. 大规模预训练模型

- **大规模预训练模型流程**



CodeGeeX模型架构

- CodeGeeX：基于GPT[1]架构的自回归模型，使用自监督学习预训练：

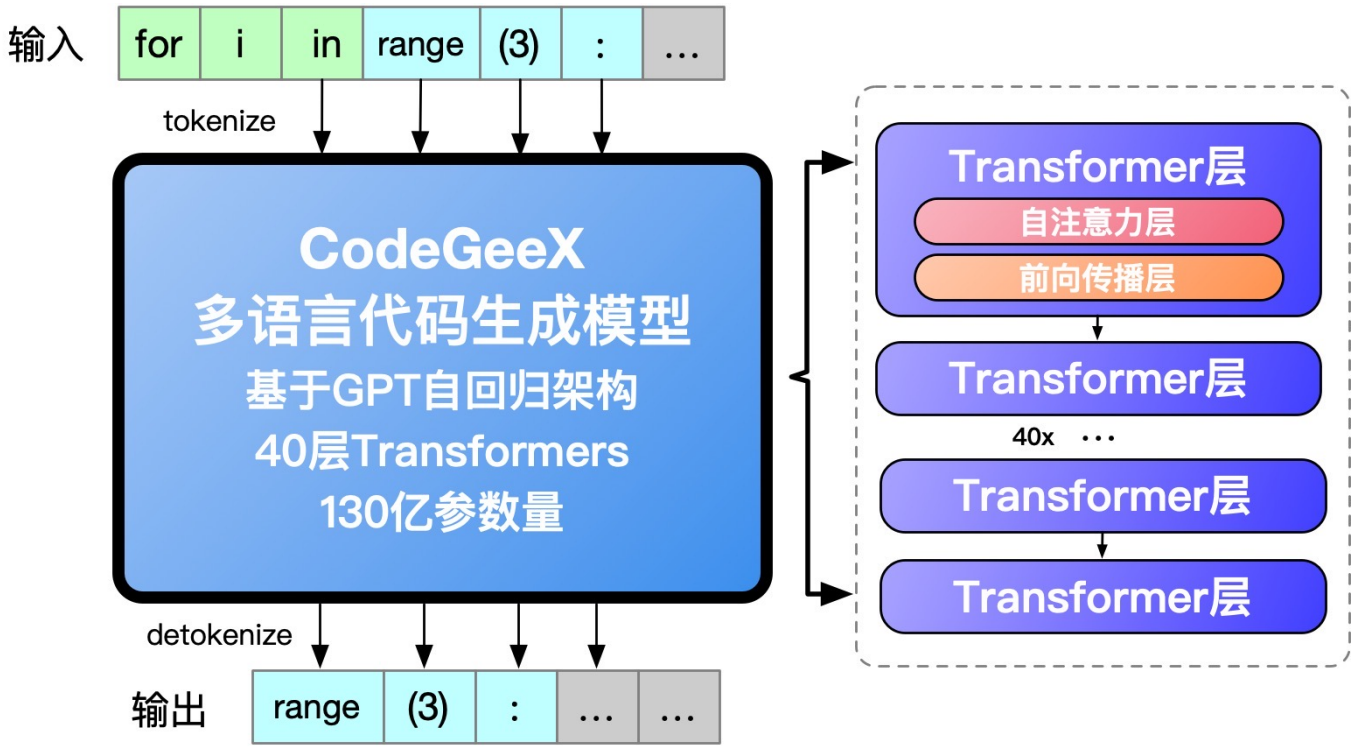


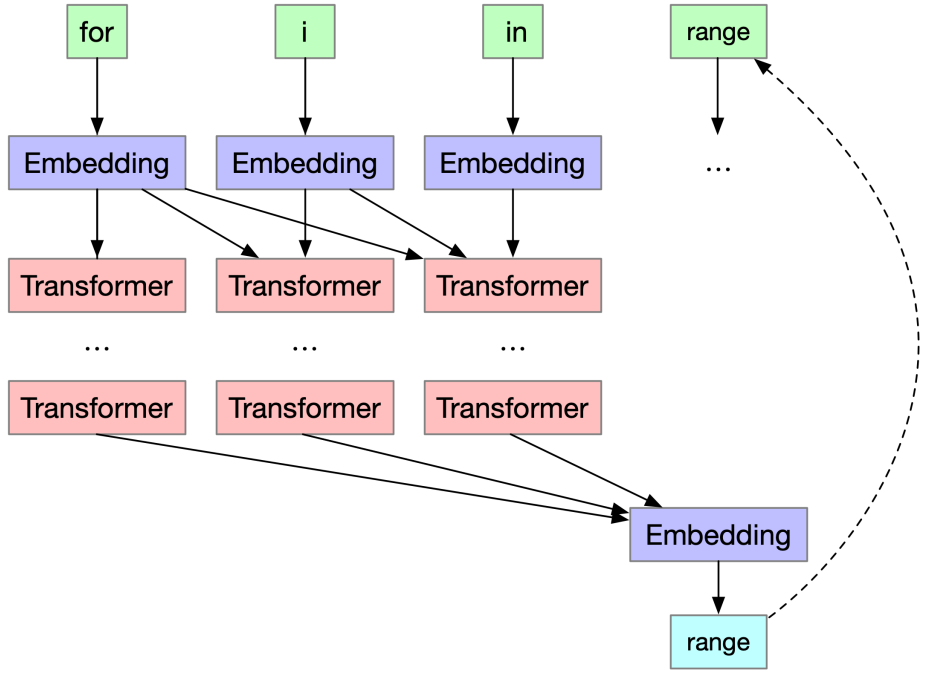
GPT（Generative Pretraining）生成式预训练示意图

Improving Language Understanding by Generative Pre-Training. (OpenAI, 2018)

CodeGeeX模型架构

- CodeGeeX：基于GPT[1]架构的自回归模型，由40层transformer[2]组成，总计参数量达130亿。
- 使用自然语言或代码token作为输入，输出下一个token的概率，支持各种编程语言相关的下游任务，如代码生成、代码补全、代码翻译、代码注释等。





使用无监督的自回归预训练

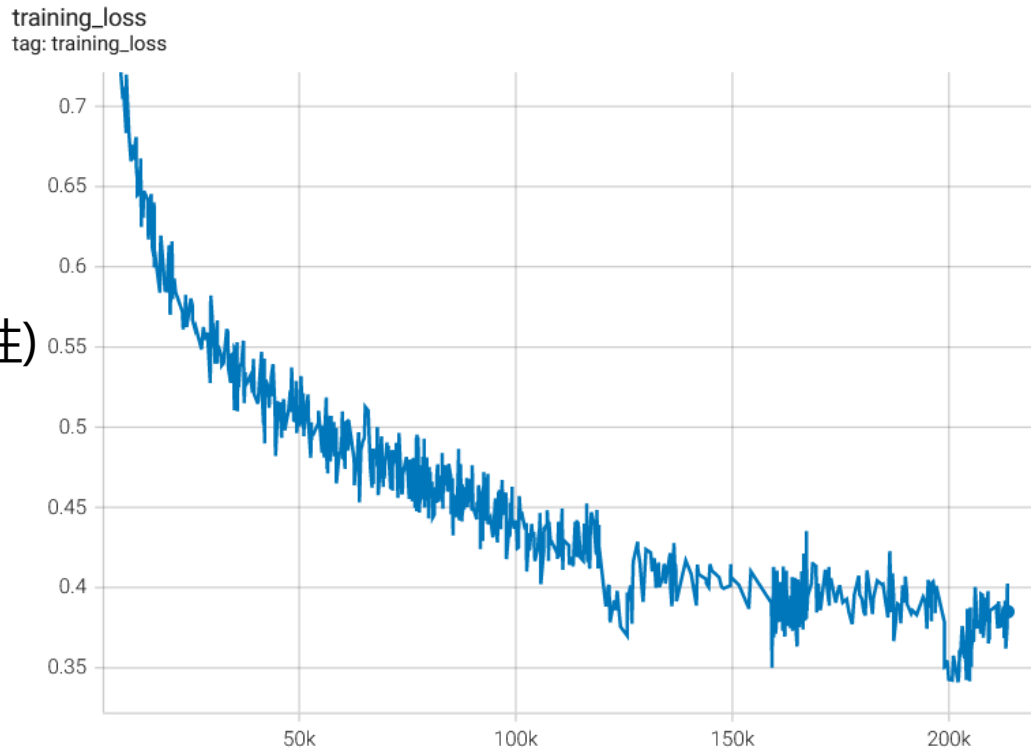
[1] Radford A, Wu J, Child R, et al. Language models are unsupervised multitask learners[J]. OpenAI blog, 2019, 1(8): 9.
[2] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

CodeGeeX: 开源的大规模多语言代码生成模型

- 大规模代码数据收集
- CodeGeeX模型架构
- **CodeGeeX模型训练及优化**
  - 基于华为Mindspore框架实现，使用昇腾910AI处理器训练
  - 进行算子融合、并行训练等优化，大幅提升训练效率
- CodeGeeX模型评估
- CodeGeeX自动编程插件
- CodeGeeX开源计划

CodeGeeX模型训练及优化

- **框架**：基于华为Mindspore 1.7
- **计算资源**：1536张昇腾910AI处理器
- **混合精度**：FP16 (Layernorm，Softmax使用FP32保证稳定性)
- **并行训练**：192路数据并行 + 8路模型并行
- **全局批大小**：3072
- **训练时长**：两个月
- **训练量**：~8500亿tokens

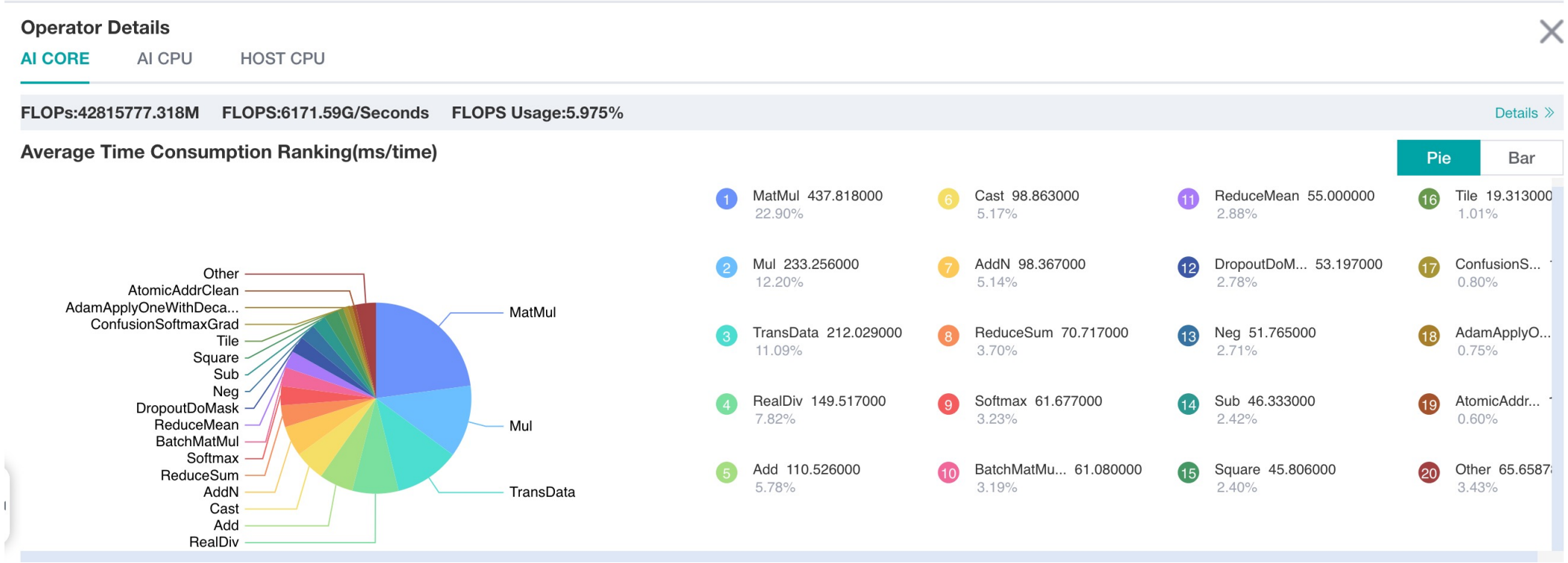


CodeGeeX训练loss下降曲线

Model	# Params	Dataset	Size	Major languages	Trained Tokens
GPT-J	6B	The Pile	Code: 96GB Natural language: 730GB	JavaScript, Java, C++, Go, C, Python	~400B
GPT-NeoX	20B	The Pile	Code: 96GB Natural language: 730GB	JavaScript, Java, C++, Go, C, Python	~472.5B
InCoder	6.7B	Public code, StackOverflow	Code: 159GB StackOverflow: 57GB	Python, JavaScript, HTML, C, C++, Java	Unknown
CodeGen -Multi	6.1B, 16.1B	The Pile, BigQuery	Code: 150.8B tokens Natural language: 354.7B tokens	Java, C++, C, Python, JavaScript, Go	~1000B (~300B for code)
CodeGeeX (Ours)	13B	The Pile (GitHub subset), CodeParrot, Other public code	Code: 158.7B tokens	C++, Python, Java, JavaScript, C, Go, HTML	~850B (all for code)

CodeGeeX模型训练及优化

- 与华为Mindspore团队合作优化
- **优化策略：**
  - 算子融合（Element-wise算子计算效率低，Layernorm/Gelu/BatchMatmul+Add）；
  - 矩阵乘算子自动搜索出效率最高的计算维度组合；



Transformer架构中Element-wise算子多，影响昇腾910计算效率

CodeGeeX模型训练及优化

- **性能提升：**
  - 单张Ascend 910训练效率由NVIDIA A100的16.7%提升至**43%**；

	优化前		优化后	
设备	NVIDIA A100	Ascend 910	NVIDIA A100	Ascend 910
卡数	160	1024	128	128
训练框架	Megatron	Mindspore 1.5	Megatron	Mindspore 1.7
并行策略	数据并行	模型并行 + 数据并行	数据并行	模型并行 + 数据并行
序列长度	2048	2048	2048	2048
全局批大小	1920	1024	512	512
单步迭代时间	30s	15s	6.5s	15s
整体训练效率	11.3B token/天	12.1B token/天	13.9B token/天†	6.0B token/天

† 环境较之前变化，通信效率更高

CodeGeeX模型训练及优化

- **性能提升：**
  - 进一步加入流水线并行后，单张Ascend 910训练效率较NVIDIA A100提升至**65%**；
  - 在千卡规模下， Ascend 910训练效率较最初提升**300%**；
  - 流水线并行本次项目并未实装，但体现了国产平台的快速迭代能力和强大竞争力；

	进一步优化后对比		进一步优化后对比（千卡规模）	
设备	NVIDIA A100	Ascend 910	Ascend 910	Ascend 910
卡数	160	128	1536	1536
训练框架	Megatron	Mindspore 1.7	Mindspore 1.7	Mindspore 1.7
并行策略	数据并行	模型并行 + 流水线并行	模型并行 + 数据并行	数据并行 + 流水线并行
序列长度	2048	2048	2048	2048
批大小	1920	1024	3072	4608
单步迭代时间	30s	20s	10s	9.7s
整体训练效率	11.3B token/天	9.1B token/天	54.3B token/天	84.1B token/天

CodeGeeX: 开源的大规模多语言代码生成模型

- 大规模代码数据收集
- CodeGeeX模型架构
- CodeGeeX模型训练及优化
- **CodeGeeX模型评估**
  - 如何正确评估代码生成的性能？
  - HumanEval-X: 新的多语言代码生成基准
- CodeGeeX自动编程插件
- CodeGeeX开源计划

如何评估代码生成模型的性能？

语义相似性 vs. 功能正确性

常见的多语言代码基准CodeXGLUE， XLCoST
均使用CodeBLEU/BLEU作为评价指标

```python
def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """ Check if in given list of numbers, are any two numbers closer to each other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    for idx, elem in enumerate(numbers):
        for idx2, elem2 in enumerate(numbers):
            if idx != idx2:
                distance = abs(elem - elem2)
                if distance < threshold:
                    return True

    return False
```

```python
def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """ Check if in given list of numbers, are any two numbers closer to each other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    for idx, elem in enumerate(numbers):
        for idx2, elem2 in enumerate(numbers):
            if idx != idx2:
                distance = abs(elem + elem2)
                if distance < threshold:
                    return True

    return False
```

❌

错误解答（CodeBLEU=98.16, BLEU=96.09）

```python
def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """ Check if in given list of numbers, are any two numbers closer to each other than
    given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    numbers = sorted(numbers)
    return min([abs(numbers[i + 1] - numbers[i]) for i in range(len(numbers) - 1)]) < threshold
```

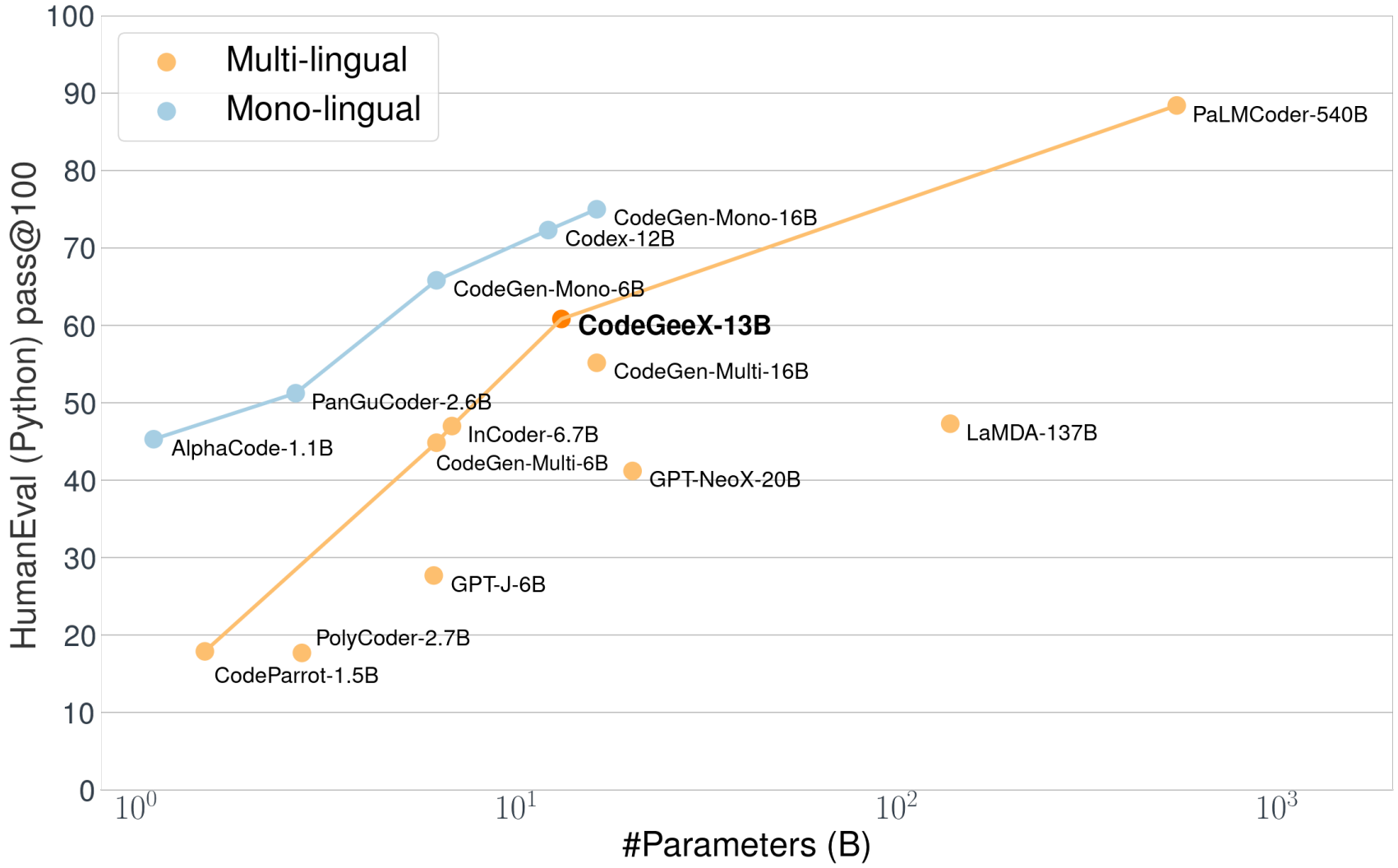
✅

正确解答（CodeBLEU=49.71, BLEU=61.45）

语义相似性并**不能正确反映**生成代码的质量，
已不满足当前评估代码生成模型的需求

如何评估代码生成模型的性能？

目前最常用的代码正确性基准HumanEval，**仅支持Python**，包含164道基础编程题，每题提供3-10个测试用例。



评价指标：

$$\text{pass@}k := \mathop{\mathbb{E}}_{\text{Problems}}\left[1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}\right]$$

一般做法是生成n=200次，估计k=1,10,100时问题被正确解答的概率。

- 多语言代码生成模型往往需要更大的模型容量，才能在单语言上取得好的效果；
- **缺乏多语言基准**，尽管模型在各种语言上训练，但仍无法评估这些语言上的效果；

HumanEval-X: 新的多语言代码生成基准

Java (Problem 0)

```java
import java.util.*;
import java.lang.*;
class Solution {
```
①

```
    /**
    Check if in given list of numbers, are any two numbers closer to each
    other than given threshold. ...
    */
```
②

```
    public boolean hasCloseElements(List<Double> numbers, double threshold) {
        for (int i = 0; i < numbers.size(); i++) {
            for (int j = i + 1; j < numbers.size(); j++) {
                double distance = Math.abs(numbers.get(i) - numbers.get(j));
                if (distance < threshold) return true;
            }
        }
        return false;
    }
}
```
③

```
public class Main {
    public static void main(String[] args) {
        Solution s = new Solution();
        List<Boolean> correct = Arrays.asList(
            s.has_close_elements(Arrays.asList(1.0, 2.0, 5.9, 4.0, 5.0), 0.95),
            !s.has_close_elements(Arrays.asList(1.0, 2.0, 5.9, 4.0, 5.0), 0.8),
                ...
        );
        if (correct.contains(false)) {
            throw new AssertionError();
        }
    }
}
```
④

Python (Problem 0)

```python
from typing import List

def has_close_elements(numbers: List[float], threshold: float) -> bool:
```
①'

```
    """
    Check if in given list of numbers, are any two numbers closer to
    each other than given threshold. ...
    """
```
②'

```
    for idx, elem in enumerate(numbers):
        for idx2, elem2 in enumerate(numbers):
            if idx != idx2:
                distance = abs(elem - elem2)
                if distance < threshold:
                    return True
    return False
```
③'

```
def check(candidate):
    assert candidate([1.0, 2.0, 5.9, 4.0, 5.0], 0.95) == True
    assert candidate([1.0, 2.0, 5.9, 4.0, 5.0], 0.8) == False
    ...

check(has_close_elements)
```
④'

HumanEval-X tasks (evaluated by functional correctness)

Generation: ① ② → ③ (Test: ① ② ③ ④)

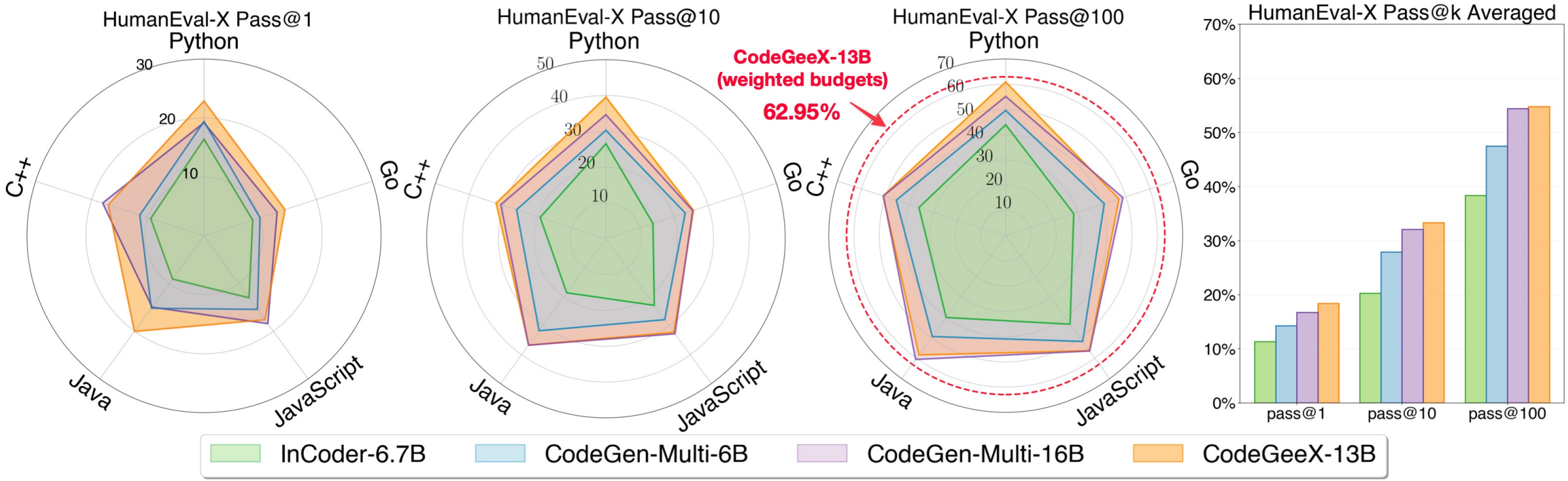
Translation: ① ③ ①' → ③' (Test: ①' ②' ③' ④')

① Declaration ② Docstring ③ Solution ④ Test

- **支持语言：**
  - Python
  - C++
  - Java
  - JavaScript
  - Go
- **数据量：**
  - 164*5=820
- **自动化测试：**
  - 编写标准解答，确保自动化测试正确性；
  - 提供脚本和测试环境；
- **下游任务：**
  - 多语言代码生成
  - 跨语言代码翻译

HumanEval-X: 代码生成任务

基线模型：InCoder-6.7B (Meta)[1], CodeGen-Multi-6B, CodeGen-Multi-16B (Salesforce)[2]

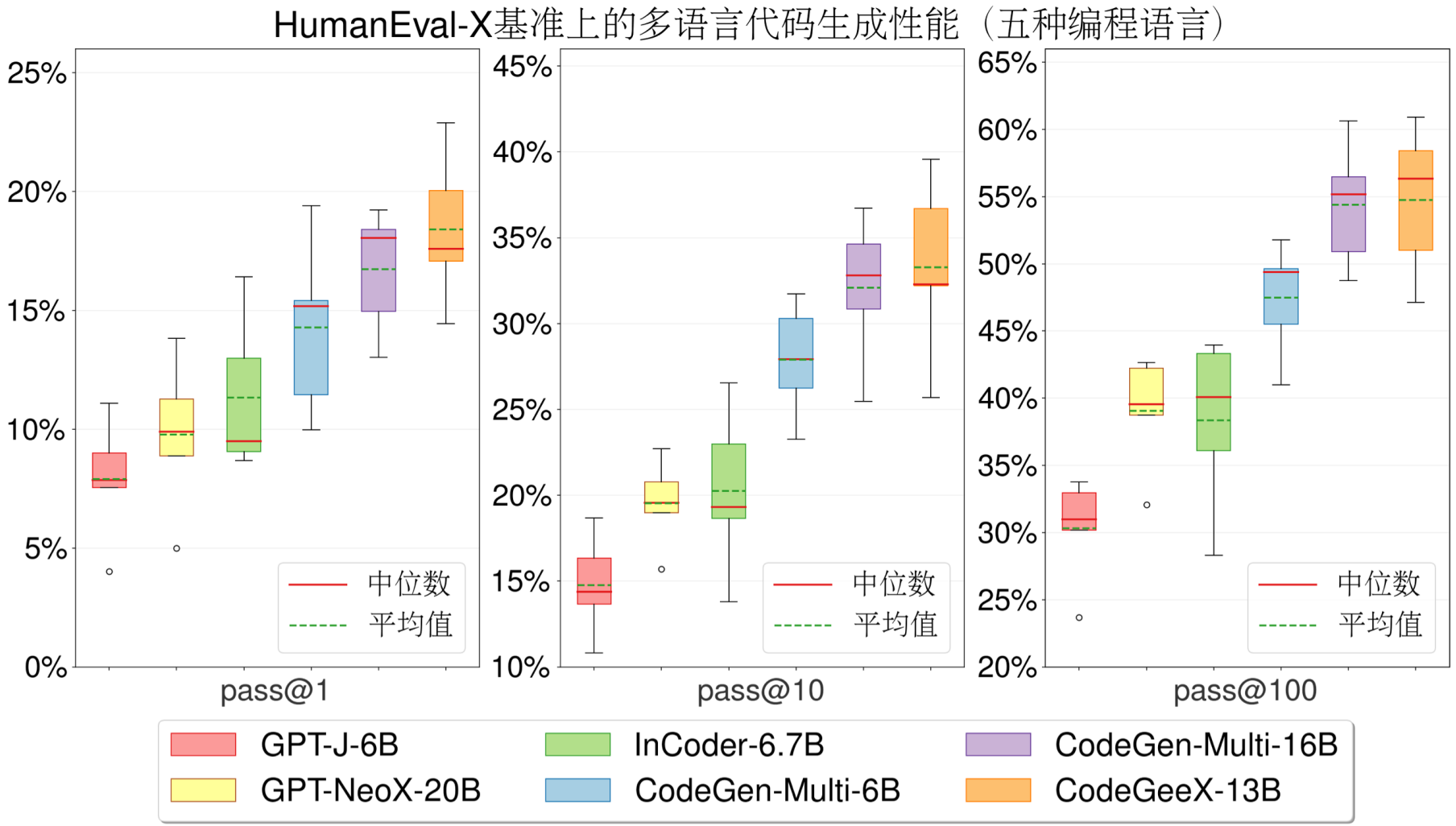


- 使用不同temperature(t=0.2/0.8) 和 nucleus sampling(top-p=0.9/0.95) 方法采样生成200次;
- CodeGeeX模型在Pass@1和Pass@10上有优势，并取得了最佳的平均性能；
- 不同模型擅长语言不同，CodeGeeX擅长Python，CodeGen-Multi-16B擅长Java；
- 分配budget到不同语言上（weighted budgets），超过单语言任一单语言的最佳性能；

[1] Daniel Fried, Armen Aghajanyan, Jessy Lin, Sida Wang, Eric Wallace, Freda Shi, Ruiqi Zhong, Wen-tau Yih, Luke Zettlemoyer, and Mike Lewis. *InCoder: A generative model for code infilling and synthesis.* arXiv preprint arXiv:2204.05999, 2022.
[2] Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. *A conversational paradigm for program synthesis.* arXiv preprint arXiv:2203.13474, 2022.

HumanEval-X: 代码生成任务

基线模型：GPT-J-6B, GPT-NeoX-20B, InCoder-6.7B, CodeGen-Multi-6B, CodeGen-Multi-16B



CodeGeeX是目前平均性能最好的开源多语言代码生成模型

CodeGeeX代码生成示例

```javascript
// language: JavaScript
/* Check if in given list of numbers, are any two numbers closer to each other than
  given threshold.
  >>> hasCloseElements([1.0, 2.0, 3.0], 0.5)
  false
  >>> hasCloseElements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
  true
  */
const hasCloseElements = (numbers, threshold) => {
/////////////// Generation ///////////////
    let found = false;
    for (let n = 0; n < numbers.length - 1; n++) {
    for (let m = n + 1; m < numbers.length; m++) {
        if (Math.abs(numbers[n] - numbers[m]) < threshold) {
        found = true;
        break;
        }
    }
    if (found) {
        break;
    }
    }
    return found;
}
```

```cpp
// language: C++
/*
Check if in given vector of numbers, are any two numbers closer to each other than
given threshold.
>>> has_close_elements({1.0, 2.0, 3.0}, 0.5)
false
>>> has_close_elements({1.0, 2.8, 3.0, 4.0, 5.0, 2.0}, 0.3)
true
*/
#include<stdio.h>
#include<vector>
#include<math.h>
using namespace std;
bool has_close_elements(vector<float> numbers, float threshold){
/////////////// Generation ///////////////
    for(int i=0; i<numbers.size(); i++){
        for(int j=i+1; j<numbers.size(); j++){
            if(abs(numbers[i]-numbers[j])<=threshold){
                return true;
            }
        }
    }
    return false;
}
```

在提供详细描述和示例的情况下，模型可以更好生成符合要求的代码；

CodeGeeX代码生成示例（其他语言输入）

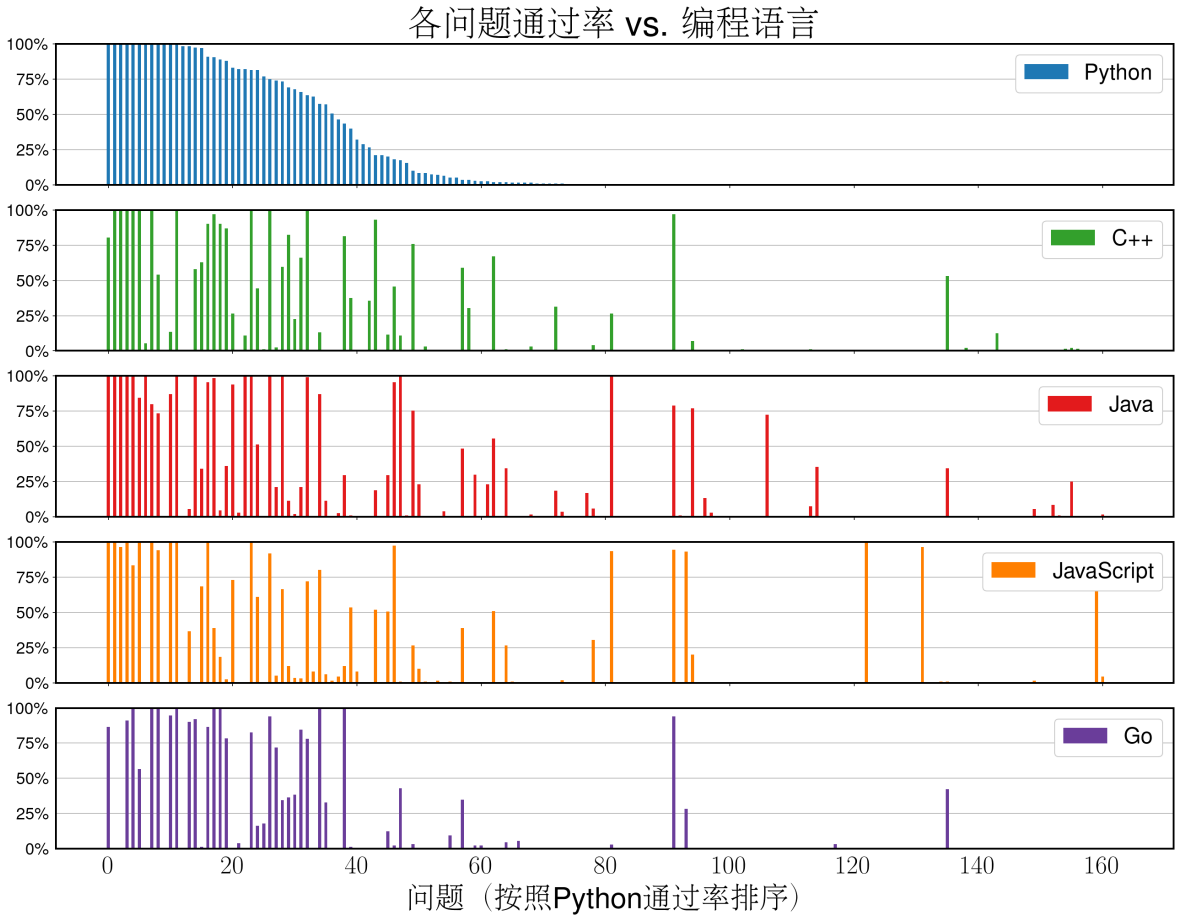
```python
# 用Python实现一个快速排序函数
################################################
def quick_sort(array):
    if len(array) <= 1:
        return array
    else:
        pivot = array[0]
        less = [i for i in array[1:] if i <= pivot]
        greater = [i for i in array[1:] if i > pivot]
        return quick_sort(less) + [pivot] + quick_sort(greater)
```

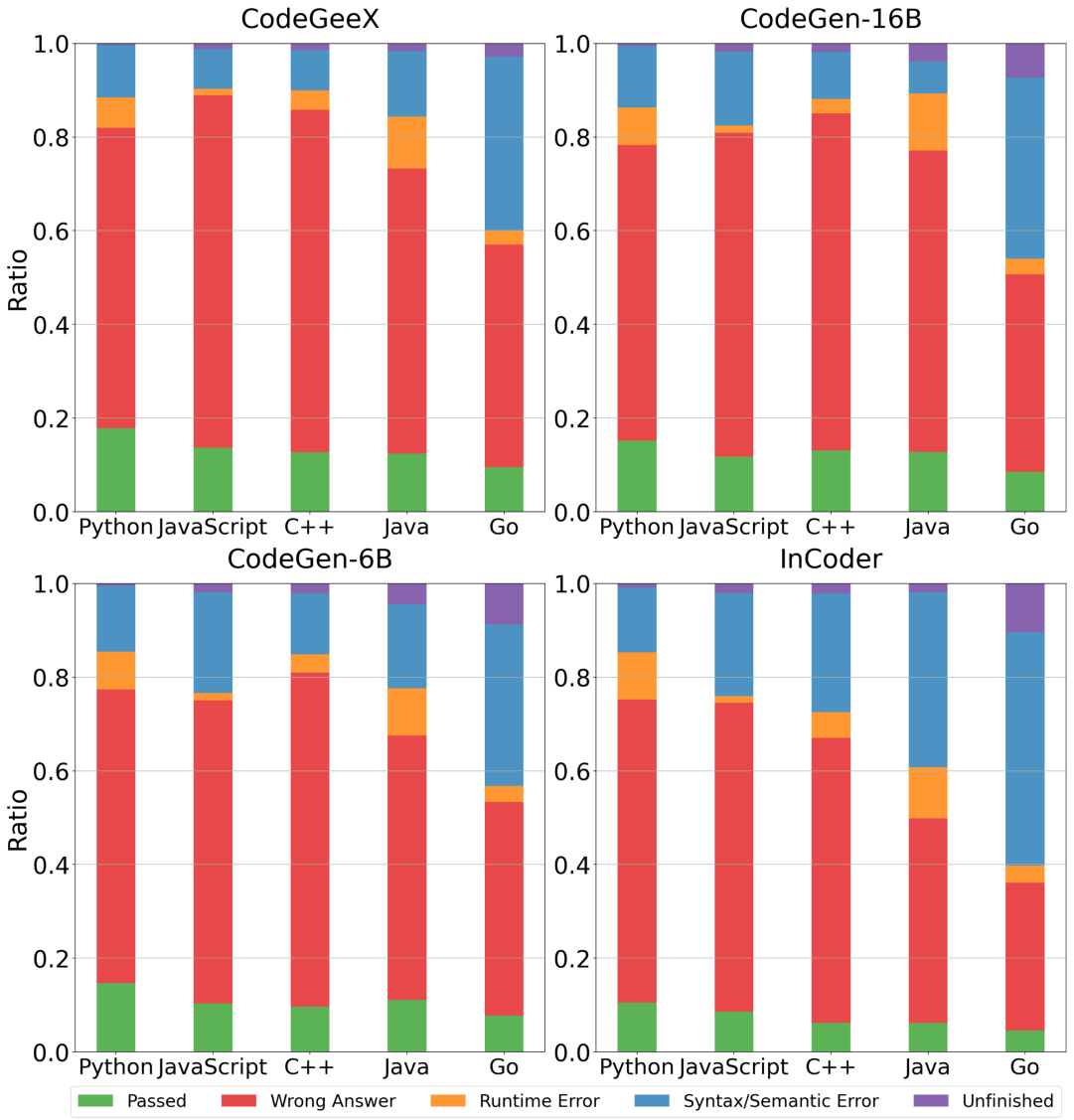
```python
# Fonction qui trie une liste dans l'ordre croissant
def tri(lst):
################################################
    """
    Tri un tableau de la liste lst en fonction de la valeur de la première ligne
    """
    for i in range(len(lst) - 1):
        for j in range(i + 1, len(lst)):
            if lst[i] > lst[j]:
                lst[i], lst[j] = lst[j], lst[i]
    return lst
```

```python
# Функция, которая сортирует список по возрастанию чисел.
################################################
def order(lst):
    # Список пуст, так как он пуст.
    if not lst:
        return lst

    # Начало цикла по первому элементу.
    # Чтобы начать отсортировать, последний элемент должен узнать последний элемент первого списка.
    for i in range(len(lst) - 1):
        for j in range(i + 1, len(lst)):
            # При сравнении элементов списка не будем искать первый элемент,
            # но будем помечать его с первым индексом слева направо.
            if lst[j] < lst[i]:
                lst[i], lst[j] = lst[j], lst[i]

    return lst
```

```java
// Javaで素数を決定する関数を作成する
################################################
public static class Main {
    public static void main(String[] args) {
        // 素数判定の例1
        System.out.println(isPrime(13)); // true
        // 素数判定の例2
        System.out.println(isPrime(19)); // false
        // 素数判定の例3
        System.out.println(isPrime(31)); // true
        // 素数判定の例4
        System.out.println(isPrime(33)); // false
    }
    private static boolean isPrime(int value) {
        // 今回は，判定なしにするために論理和が0のときは，trueを返す。
        if (value == 0) {
            return true;
        }
        // 1より大きな数字を計算しているときは，falseを返す。
        if (value == 1) {
            return false;
        }
        // 2以上の値を満たす数字があるときは，trueを返す。
        if (value < 2) {
            return true;
        }
        // 初期化しておく。
        int squareRoot = (int) Math.sqrt(value);
        // 2以上の値を満たす数字があるときは，trueを返す。
        for (int i = 2; i <= squareRoot; i++) {
            if (value % i == 0) {
                return false;
            }
        }
        // 2以上の値を満たす数字がないときは，trueを返す。
        return true;
    }
}
```

CodeGeeX代码生成结果分析

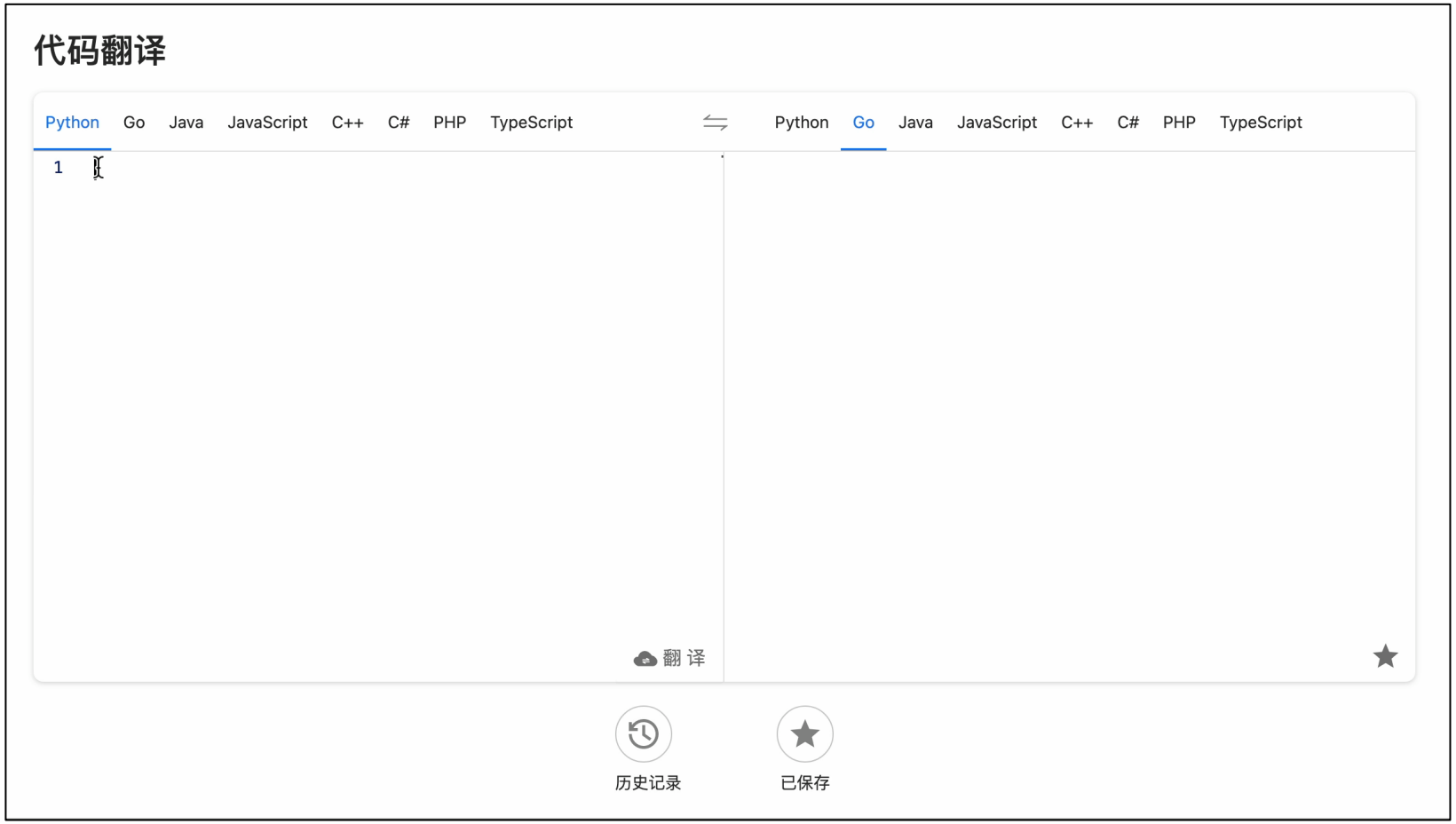




观察1: 不同语言解决问题的分布存在差异；

观察2: 语法和运行时错误占比低，主要是输出结果不对；

HumanEval-X: 代码翻译任务



把一种语言的代码翻译成另一种语言

DEMO: https://models.aminer.cn/codegeex/codeTranslator

挑战：

- 5种语言，共计20个翻译对;
- 侧重于**代码正确性**，
  不关注语义相似性；
- 不加入自然语言描述，要求模型
  从解答代码中理解题目在做什么；
- 不同语言间语法和特性差别大，
  如果逐句翻译，无法保证正确性；

HumanEval-X: 代码翻译任务

- 设计输入prompt的格式，让模型更倾向于翻译代码；
- 使用不同语言成对的代码对模型进行微调，可以显著提高翻译的效果；

	Model	Target Language: Python @1	Python @10	Python @100	C++ @1	C++ @10	C++ @100	Java @1	Java @10	Java @100	JavaScript @1	JavaScript @10	JavaScript @100	Go @1	Go @10	Go @100
Py	InCoder-6.7B	-	-	-	26.11	41.00	54.25	26.74	42.66	61.20	37.05	58.85	78.91	15.69	27.57	43.67
	CodeGen-Multi-16B	-	-	-	**35.94**	**47.81**	59.37	29.27	45.70	64.45	**43.40**	**66.26**	**82.55**	**28.87**	**41.01**	**57.72**
	CodeGeeX-13B	-	-	-	26.54	43.56	56.48	25.84	41.52	59.72	23.22	47.33	65.87	9.56	23.83	33.56
	CodeGeeX-13B-FT	-	-	-	34.16	46.86	**61.22**	**41.98**	**58.17**	**72.78**	34.81	53.05	66.08	16.41	30.76	46.37
C++	InCoder-6.7B	34.37	58.41	78.57	-	-	-	34.04	57.02	68.70	37.05	65.05	79.61	25.54	39.11	58.02
	CodeGen-Multi-16B	33.83	55.37	76.64	-	-	-	43.20	69.84	**88.82**	**54.51**	**71.50**	**83.14**	**27.94**	**49.73**	**68.32**
	CodeGeeX-13B	27.18	49.02	67.69	-	-	-	22.56	40.91	64.08	30.23	55.68	75.58	8.64	18.79	31.76
	CodeGeeX-13B-FT	**62.79**	**80.39**	**87.10**	-	-	-	**71.68**	**81.62**	85.84	50.83	64.55	74.57	16.71	34.18	52.98
Java	InCoder-6.7B	42.76	65.55	80.43	40.01	55.17	70.39	-	-	-	43.20	**68.24**	**84.39**	21.58	35.20	54.97
	CodeGen-Multi-16B	52.73	69.30	82.74	41.42	54.68	65.50	-	-	-	**57.65**	67.90	79.22	**34.00**	**48.49**	**67.94**
	CodeGeeX-13B	43.41	68.46	84.03	39.33	58.48	72.36	-	-	-	44.19	64.22	82.89	17.17	32.74	47.71
	CodeGeeX-13B-FT	**75.03**	**87.71**	**95.13**	**49.67**	**65.65**	**75.40**	-	-	-	49.95	62.82	79.64	18.85	32.92	48.93
JS	InCoder-6.7B	23.18	50.47	67.26	35.47	54.48	70.71	30.67	50.90	71.03	-	-	-	25.79	42.96	61.47
	CodeGen-Multi-16B	35.52	52.23	69.78	35.41	53.12	64.47	33.79	56.06	74.00	-	-	-	**33.38**	**49.08**	**64.14**
	CodeGeeX-13B	31.15	54.02	72.36	30.32	51.63	69.37	24.68	48.35	69.03	-	-	-	11.91	26.39	39.81
	CodeGeeX-13B-FT	**67.63**	**81.88**	**89.30**	**46.87**	**60.82**	**73.18**	**56.55**	**70.27**	**80.71**	-	-	-	16.46	32.99	50.29
Go	InCoder-6.7B	34.14	54.52	70.88	30.45	48.47	62.81	34.52	53.95	69.92	39.37	**63.63**	**80.75**	-	-	-
	CodeGen-Multi-16B	38.32	50.57	68.65	32.95	45.88	59.56	36.55	59.12	78.70	38.93	56.68	70.68	-	-	-
	CodeGeeX-13B	35.92	56.02	77.32	29.83	41.98	58.15	22.89	41.04	61.46	25.24	46.50	69.93	-	-	-
	CodeGeeX-13B-FT	**57.98**	**79.04**	**93.57**	**38.97**	**53.05**	**63.92**	**54.22**	**69.03**	**79.40**	**43.07**	59.78	74.04	-	-	-

*InCoder-6.7B使用Multi-task训练，CodeGeeX-13B-FT使用XLCoST翻译数据进行Finetune

在20个翻译对中，我们观察到A-to-B和B-to-A的性能往往是负相关的，这反映了当前的多语言代码生成模型的学习能力还不足以很好地掌握各种语言之间的联系；

CodeGeeX代码翻译示例

```
code translation
Java:
import java.util.*;
import java.lang.*;

class Solution {
    public List<Integer> sortThird(List<Integer> l) {

        List<Integer> thirds = new ArrayList<>();
        for (int i = 0; i < l.size(); i += 3) {
            thirds.add(l.get(i));
        }
        Collections.sort(thirds);
        List<Integer> result = l;
        for (int i = 0; i < l.size(); i += 3) {
            result.set(i, thirds.get(i / 3));
        }
        return result;
    }
}
JavaScript:

const sortThird = (l) => {
/////////////// Translation ///////////////
  let thirds = [];
  for (let i = 0; i < l.length; i += 3) {
    thirds.push(l[i]);
  }
  thirds.sort();
  const result = l;
  for (let i = 0; i < l.length; i += 3) {
    result[i] = thirds[i / 3];
  }
  return result;
}
```

```
code translation
Go:

func SortThird(l []int) []int {

    temp := make([]int, 0)
    for i := 0; i < len(l); i = i + 3 {
        temp = append(temp, l[i])
    }
    sort.Ints(temp)
    j := 0
    for i := 0; i < len(l); i = i + 3 {
        l[i] = temp[j]
        j++
    }
    return l
}
C++:
vector<int> sort_third(vector<int> l){
/////////////// Translation ///////////////
    vector<int> temp;
    for(int i = 0; i < l.size(); i = i + 3){
        temp.push_back(l[i]);
    }
    sort(temp.begin(), temp.end());
    int j = 0;
    for(int i = 0; i < l.size(); i = i + 3){
        l[i] = temp[j];
        j++;
    }
    return l;
}
```

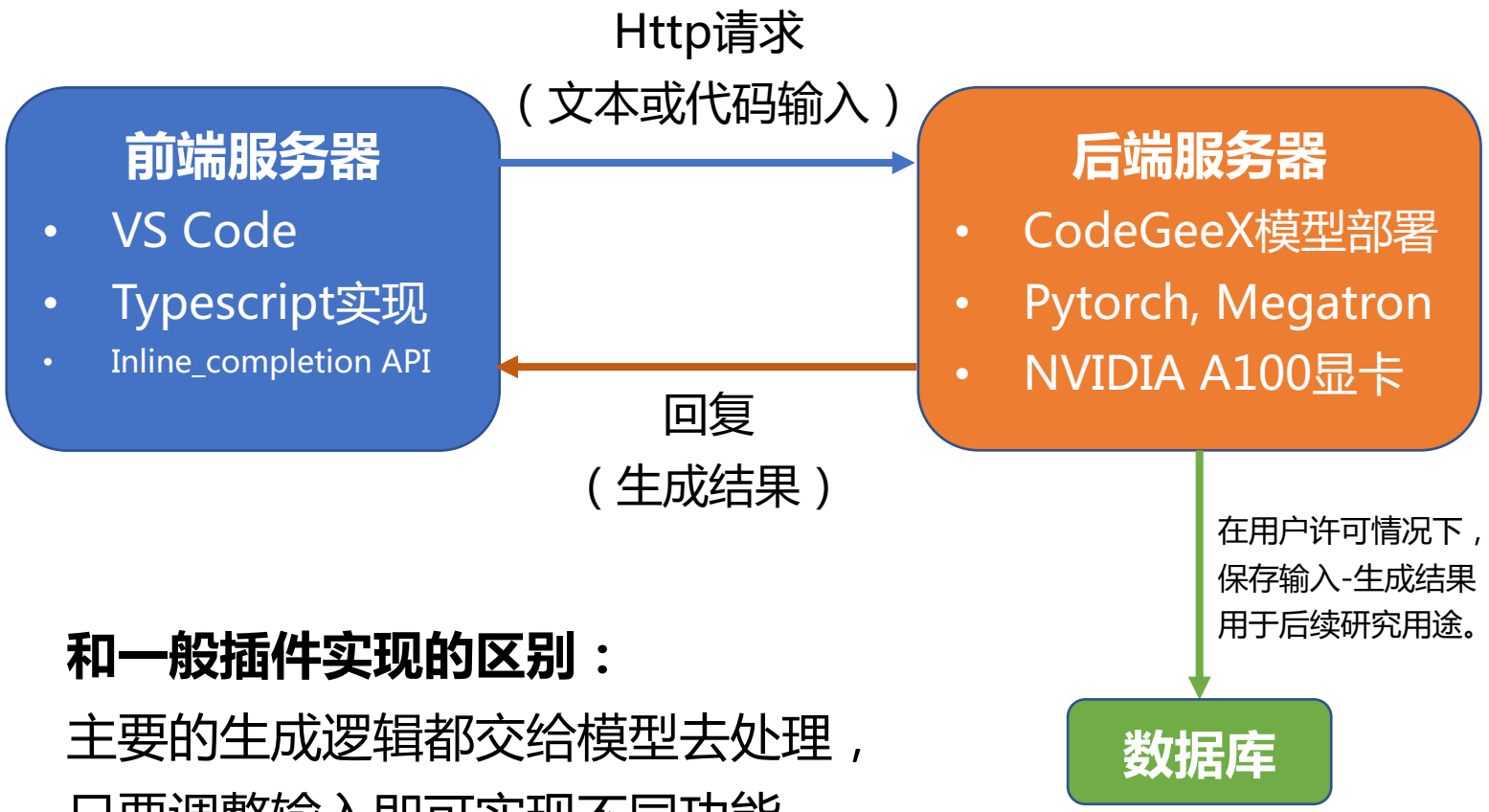
设计特别的提示格式，使模型可以进行零样本翻译。

CodeGeeX: 开源的大规模多语言代码生成模型

- 大规模代码数据收集
- CodeGeeX模型架构
- CodeGeeX模型训练及优化
- CodeGeeX模型评估
- **CodeGeeX自动编程插件**
  - VS Code平台免费下载
  - 支持代码生成、补全、翻译等功能
- CodeGeeX开源计划

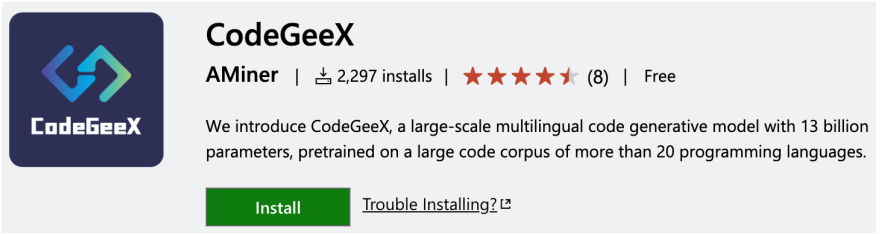
CodeGeeX自动编程插件

基于CodeGeeX，我们开发了VS Code上的自动编程插件，提供多种交互模式，支持代码生成、补全、翻译、注释等功能，免费使用，更好辅助程序员开发。



和一般插件实现的区别：

主要的生成逻辑都交给模型去处理，只要调整输入即可实现不同功能。



VS Code插件市场搜索“codegeex”免费下载

目前支持的**四种主要模式**：

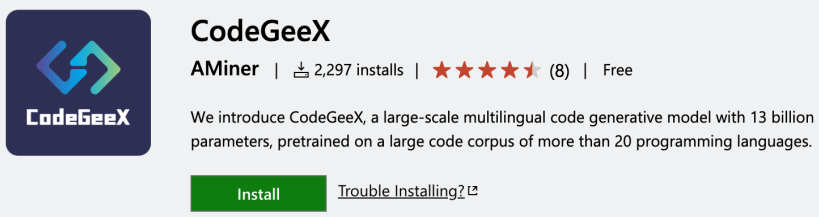
1. 补全模式
2. 交互模式
3. 翻译模式（未正式上线）
4. 提示模式

CodeGeeX自动编程插件

example_python.py 1, U

```python
import torch

class SelfAttention(torch.nn.Module):
    """self-attention layer abstract class.
    Self-attention layer takes input with size [b, s, h]
    and returns output of the same size.
    """
    def __init__(self, hidden_size, num_attention_heads, layer_number):
```

main* 1 0 Ln 8, Col 72 Spaces: 4 UTF-8 LF Python 3.10.8 64-bit



VS Code插件市场搜索“codegeex”免费下载

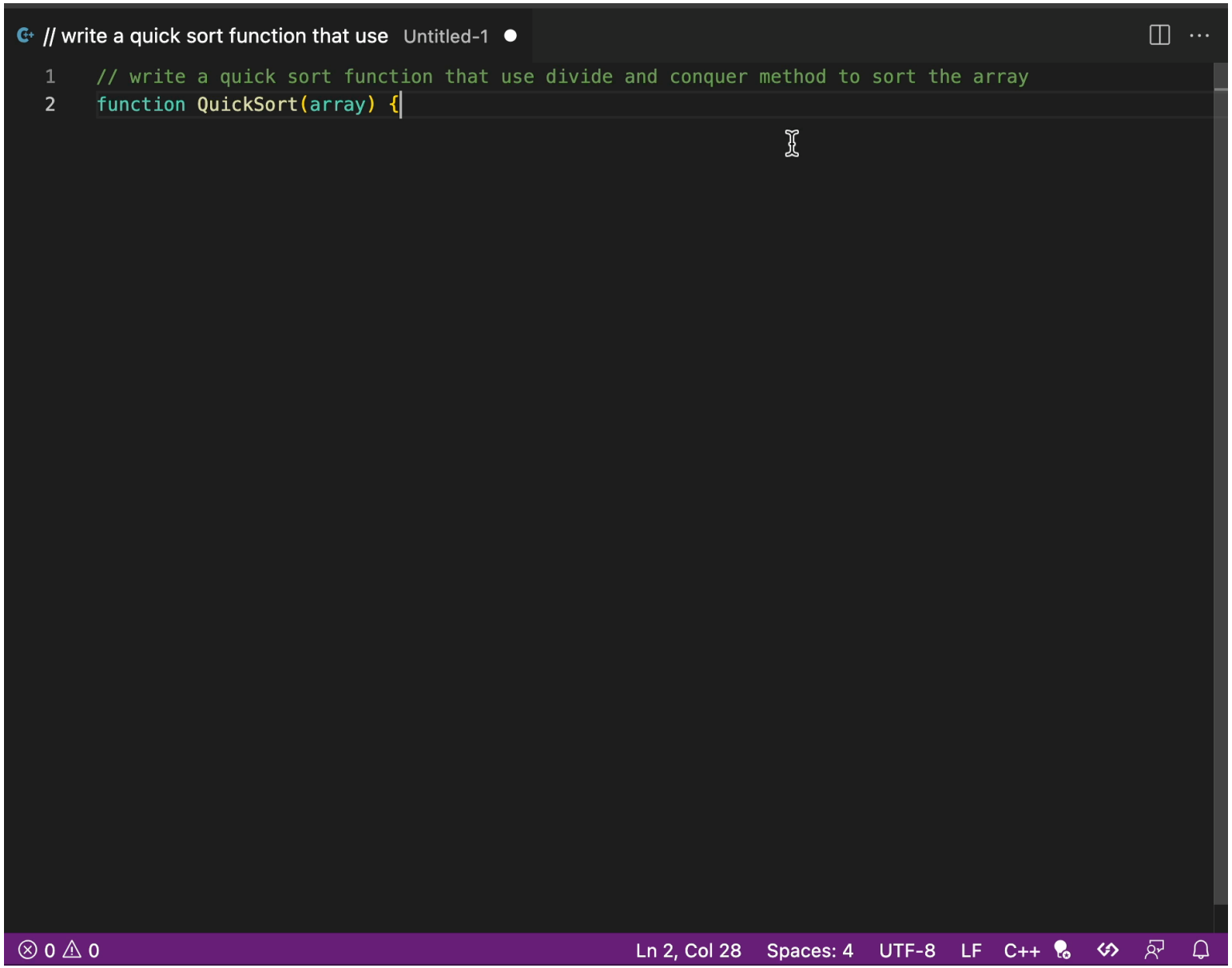
基于CodeGeeX，我们开发了VS Code上的自动编程插件，提供多种交互模式，支持代码生成、补全、翻译、注释等功能，免费使用，更好辅助程序员开发；

9月20日上线以来，CodeGeeX插件服务了**3500+**用户，累计调用量超过**300万次**；

补全模式：

在当前光标处自动提示候选代码

CodeGeeX自动编程插件

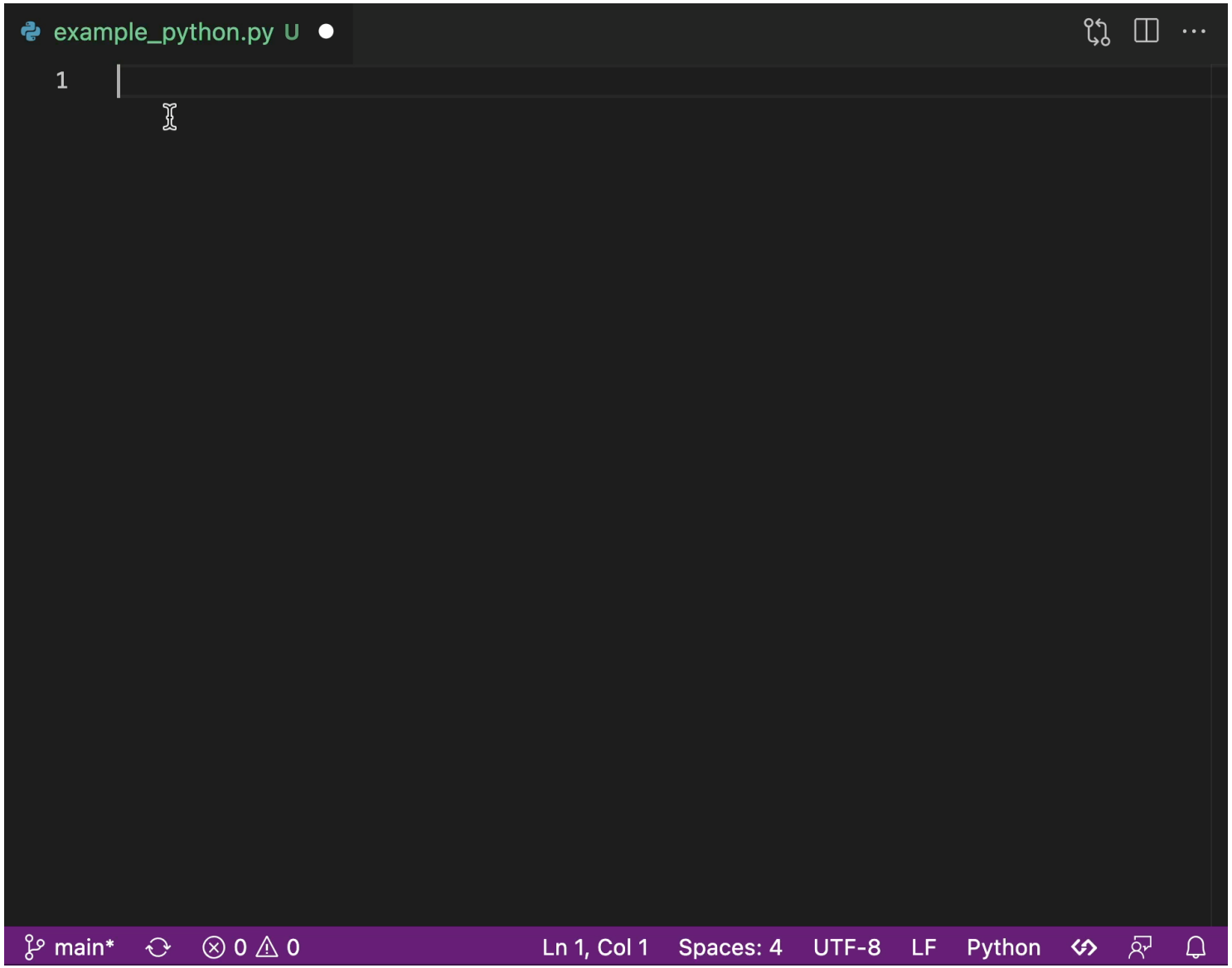


交互模式：

按Ctrl+Enter进入交互模式，

可在右侧窗口中选择生成结果

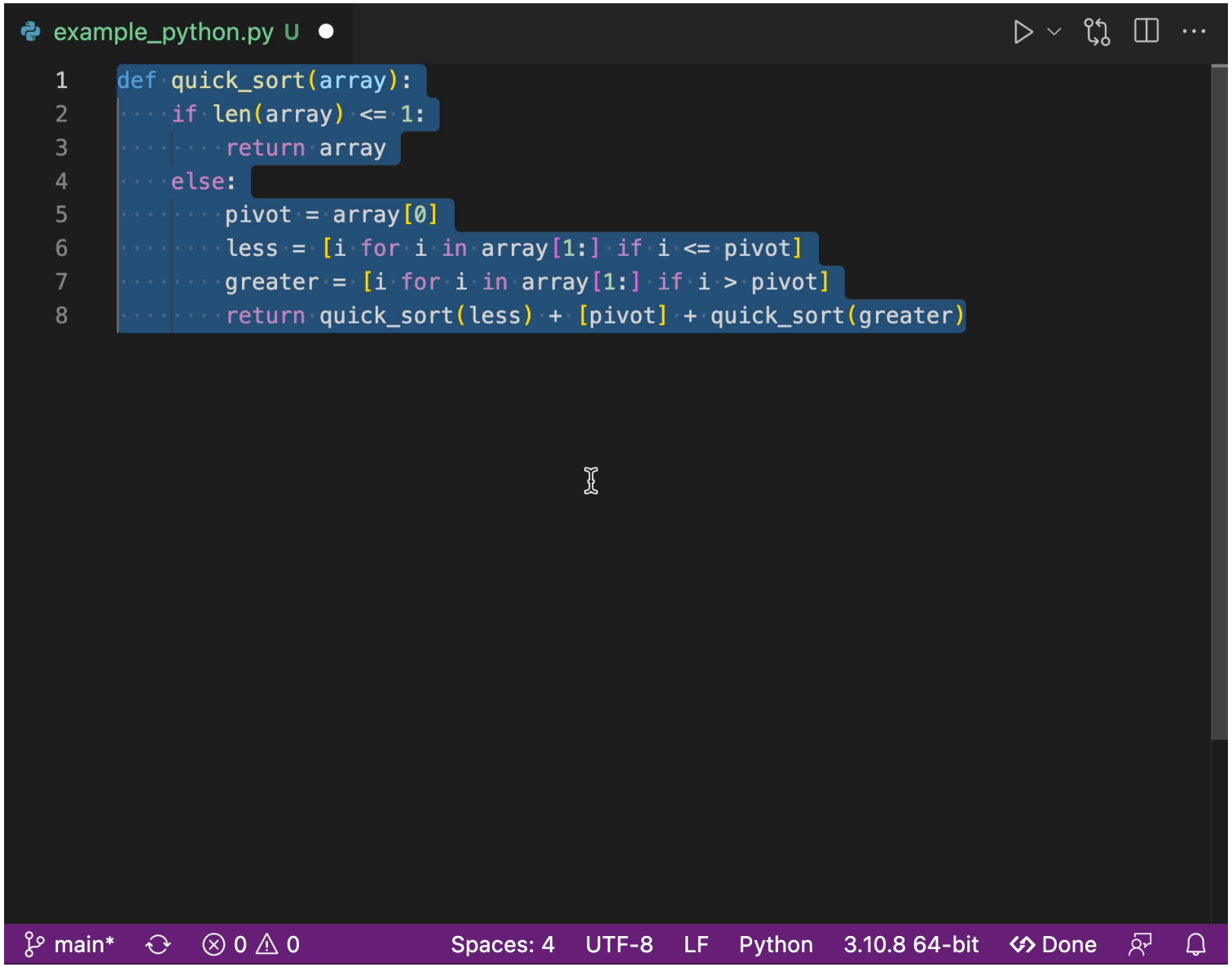
CodeGeeX自动编程插件



翻译模式：

贴入一段待翻译代码，Ctrl+Alt+T激活翻译模式，您根据提示选择该代码语言，然后CodeGeeX会匹配到当前编辑器语言的代码。点击翻译结果上方的``use code``即可插入。可以在设置中选择注释或者覆盖原来的代码。

CodeGeeX自动编程插件




```python
# language: Python

def sum_squares(lst):
    sum = 0
    for i in range(len(lst)):
        if i % 3 == 0:
            lst[i] = lst[i]**2
        elif i % 4 == 0:
            lst[i] = lst[i]**3
        sum += lst[i]
    return sum

<INPUT>

# Explain the code line by line
def sum_squares(lst):
    # initialize sum
    sum = 0
    # loop through the list
    for i in range(len(lst)):
        # if the index is a multiple of 3
        if i % 3 == 0:
            # square the entry
            lst[i] = lst[i]**2
        # if the index is a multiple of 4
        elif i % 4 == 0:
            # cube the entry
            lst[i] = lst[i]**3
        # add the entry to the sum
        sum += lst[i]
    # return the sum
    return sum

# Explain the code line by line
<INPUT:0,1>
```

提示模式：

预先写好自定义模板文件（如上），

在输入中加入额外提示例子，

按Option/Alt+T进入提示模式，

选择特定模板生成代码；

CodeGeeX: 开源的大规模多语言代码生成模型

- 大规模代码数据收集
- CodeGeeX模型架构
- CodeGeeX模型训练及优化
- CodeGeeX模型评估
- CodeGeeX自动编程插件
- **CodeGeeX开源计划**
  - 未来的开源开发计划
  - 更快+更好+更多应用场景

CodeGeeX开源开发计划

CodeGeeX
开源仓库

我们已经开源了这些内容：

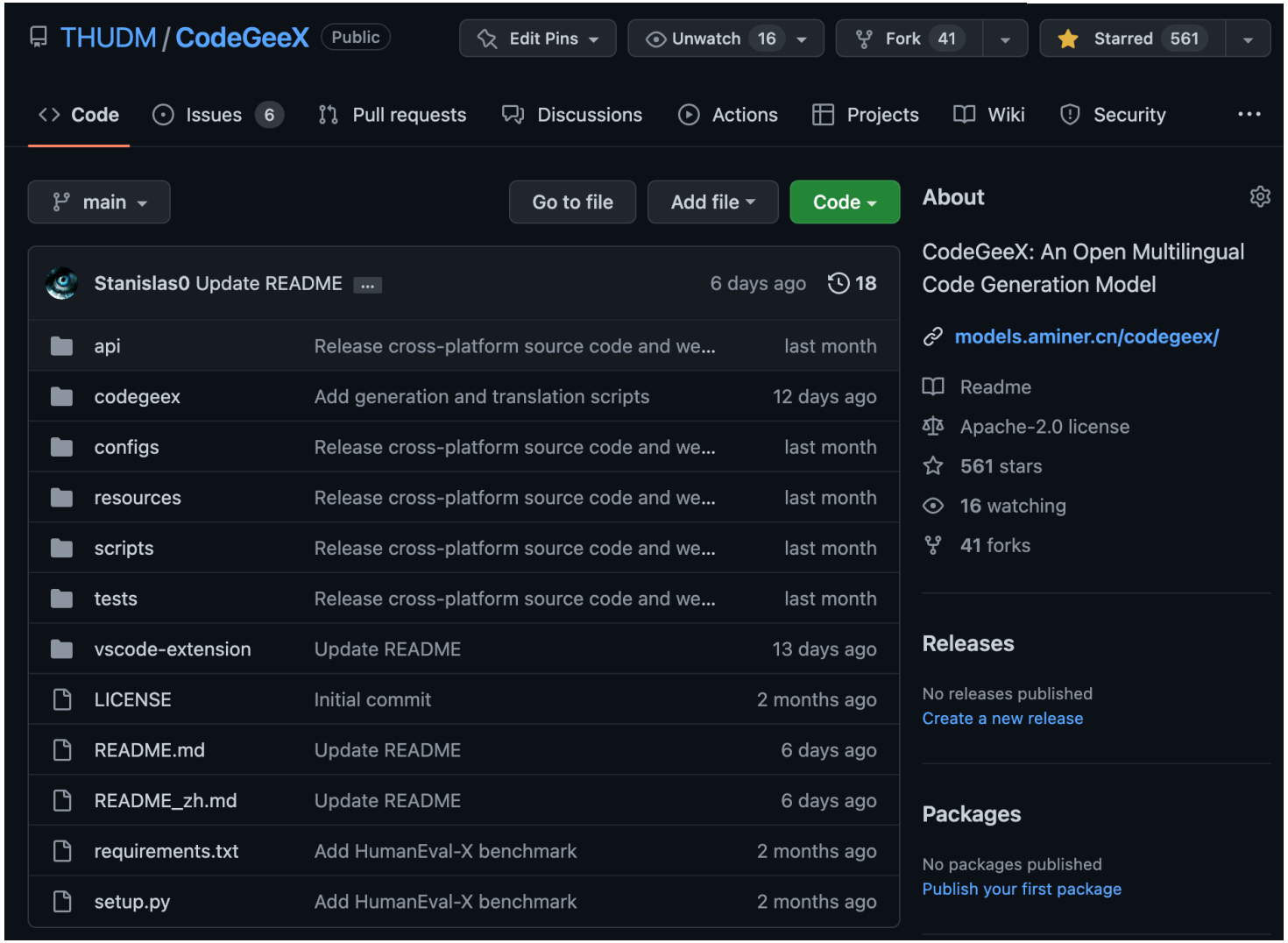
- 跨平台模型代码（支持昇腾、英伟达）
- 模型训练、微调、推理、测评代码
- 模型权重（可在官网申请下载）
- HumanEval-X数据集

未来还有更多：

- 各个平台的插件代码
- 模型加速、量化代码
- 微调迭代后的模型权重
- 。。。

希望有更多开发者加入我们，

一起让CodeGeeX变得更好！



CodeGeeX开源开发计划

目前亟待开发的功能：

1. **多平台支持**
    1. Jetbrains（正在开发中）、Xcode、Vim/NeoVim、Eclipse and more！
2. **提示模式通用语法设计**
    1. 支持在模板中加入<Input>, <Output:0,1>这类语法来实现各种不同功能；
    2. 后端可以正确解析该语法；
    3. 丰富模板库；
3. **优化交互体验**
    1. 是否有更加友好的交互方式

CodeGeeX开源开发计划

CodeGeeX有着无限的潜力，这里我们列出**十个应用场景+十个待解决的技术问题：**

十个应用场景：

1. 前端代码生成
2. 操作系统辅助开发
3. 形式化数学定理证明
4. 自动化bug提示与修复
5. 自动化代码审查
6. 代码注释和概括
7. 不同编程语言项目迁移
8. 青少年编程教育辅助
9. 高效的游戏开发
10. 更智能的下一代IDE

。。。

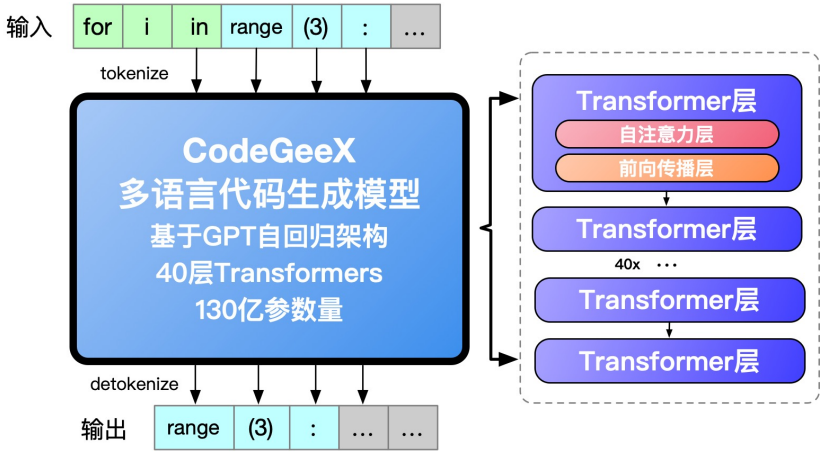
十个待解决技术问题：

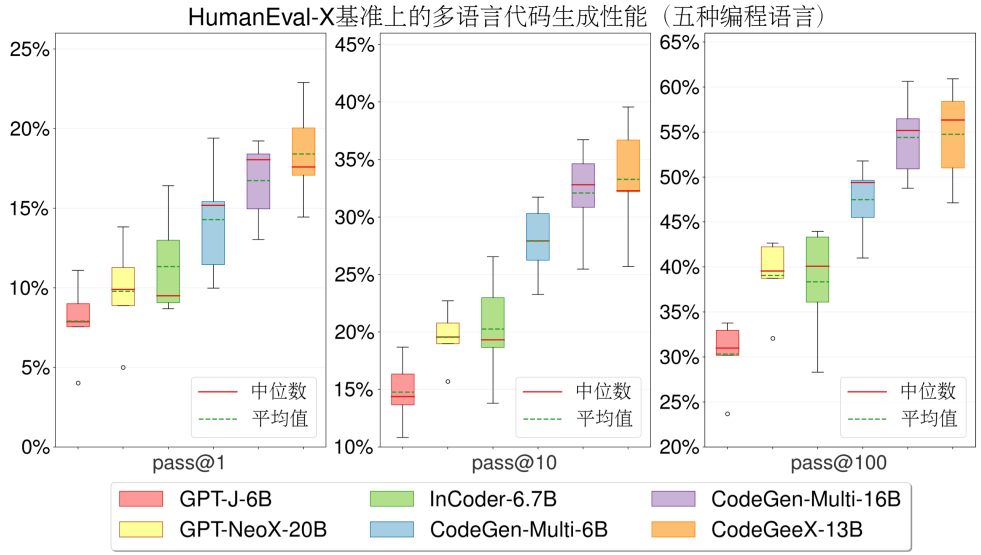
1. 超轻量化模型
2. 更智能的解码策略
3. 更符合人类指令的输出
4. 生成代码溯源问题
5. 超长代码生成
6. 跨文件代码生成
7. 生成代码的评估
8. 根据编译器反馈优化生成
9. 检索增强代码生成
10. 更优的代码生成微调范式

。。。

欢迎研究者、软件开发者、开源爱好者们和我们一起探索更多的可能性！！！

CodeGeeX: 开源的大规模多语言代码生成模型





CodeGeeX主页

CodeGeeX是一个具有**130亿**参数的多编程语言代码生成预训练模型，支持十多种编程语言。开源开放，同时支持昇腾和英伟达平台，具有高精度代码生成、代码翻译等能力。基于CodeGeeX，我们开发了免费的自动代码生成插件，致力于提高广大程序员的编程效率！

感谢鹏城实验室对本项目的算力支持！

感谢智谱AI、华为Mindspore团队对本项目的技术支持！

清华大学
Tsinghua University

Q&A

体验DEMO

GitHub仓库

主页: https://models.aminer.cn/codegeex

邮箱: codegeex@aminer.cn

源码: https://github.com/THUDM/CodeGeeX

插件: VS Code插件市场搜索“codegeex”免费下载

欢迎加入「CodeGeeX 开发者交流群」

自动加群方法

> 添加小呆个人微信号好友

扫描下方二维码或添加微信号“atombot2err”发送好友申请，内容填“CodeGeeX”，小呆会自动通过并邀请您进群。

（受微信官方限制，若好友申请没有立即通过，请稍等片刻再尝试）

▲ 好友添加示意图

▲ 小呆个人微信号二维码

> 如果您已经是小呆好友

直接给小呆发送“CodeGeeX”即可

▲ 对话内容示意图