Cangjie Community

仓颉语言 KOL 交流会

仓颉语言库开发经验分享

主讲人：杨海龙 博士在读

江南大学人工智能与计算机学院

把握仓颉新机　蓄势引领辉煌

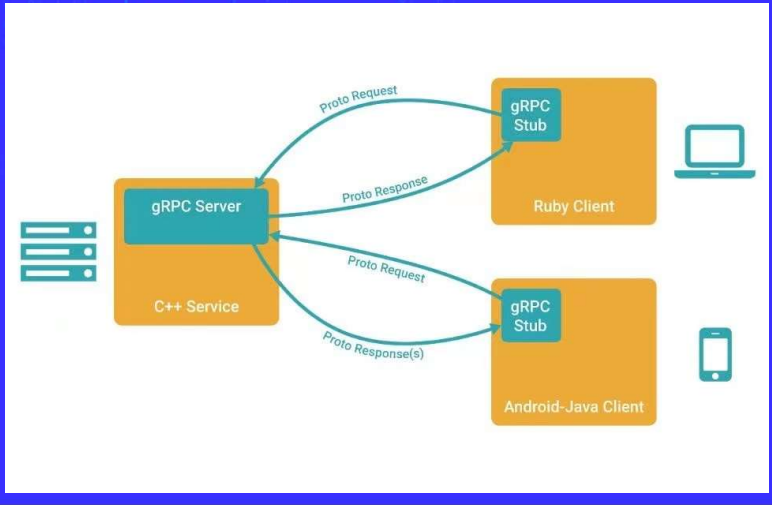
目录

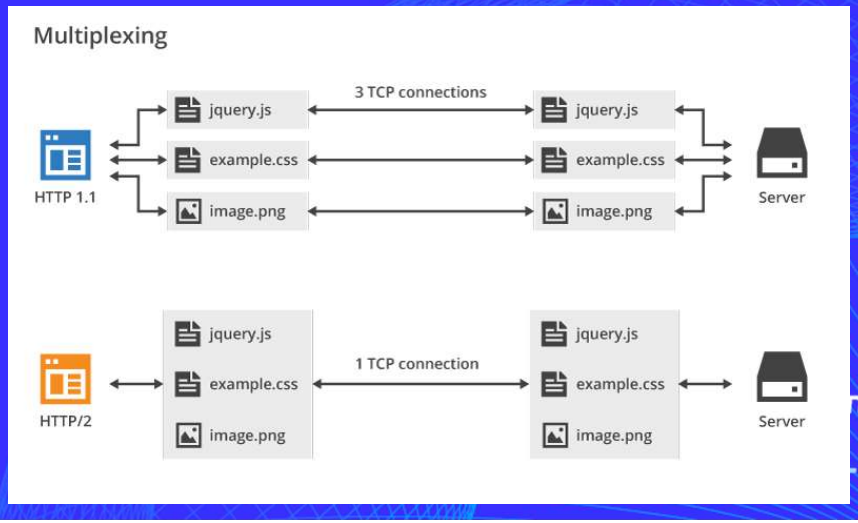
仓颉与网络编程

仓颉网络库，官方库Http1.1、TLS 

社区库Http2.0、rpc4cj 

- **Https**协议提高了安全性，用户登录、在线支付、防止篡改、保护隐私。特别是开发webapi，需要基于https协议。
- **GPRC**协议是基于Http2.0，仓颉社区项目rpc4cj。rpc4cj 可以跨语言、跨端调用接口，特别适合多语言支持、多端适配的项目。





仓颉与解析器

仓颉文件解析器，主要是仓颉社区贡献，包括csv4cj、yaml4cj、toml4cj、xml4cj、dom4cj、html4cj、kv4cj等。

- 如何对文件进行分词获得相应字段的关键词、数值等的token?
- 如何将这些token转化成仓颉的整数、浮点、布尔、字符、字符串等类型?

根据文件的格式和语法，将原文将切分成类似字符串的token。关于这些token，仓颉并不能识别其真实的数据类型，需要通过领域知识赋予**仓颉类型**给这些token，例如“title”是类似key的一个token，仓颉数据类型便是字符串。

建议仓颉解析器输出数据是**json类型**，可以通过官方的**encoding模块**与其他格式的数据互转。

源文件

```
# This is a TOML document
 title = "TOML Example"
[owner]
name = "Tom"
age = 19
```

分词

```
title
=
"TOML Example"
[
owner
]
name
=
"Tom"
age
 =
19
```

赋予类型

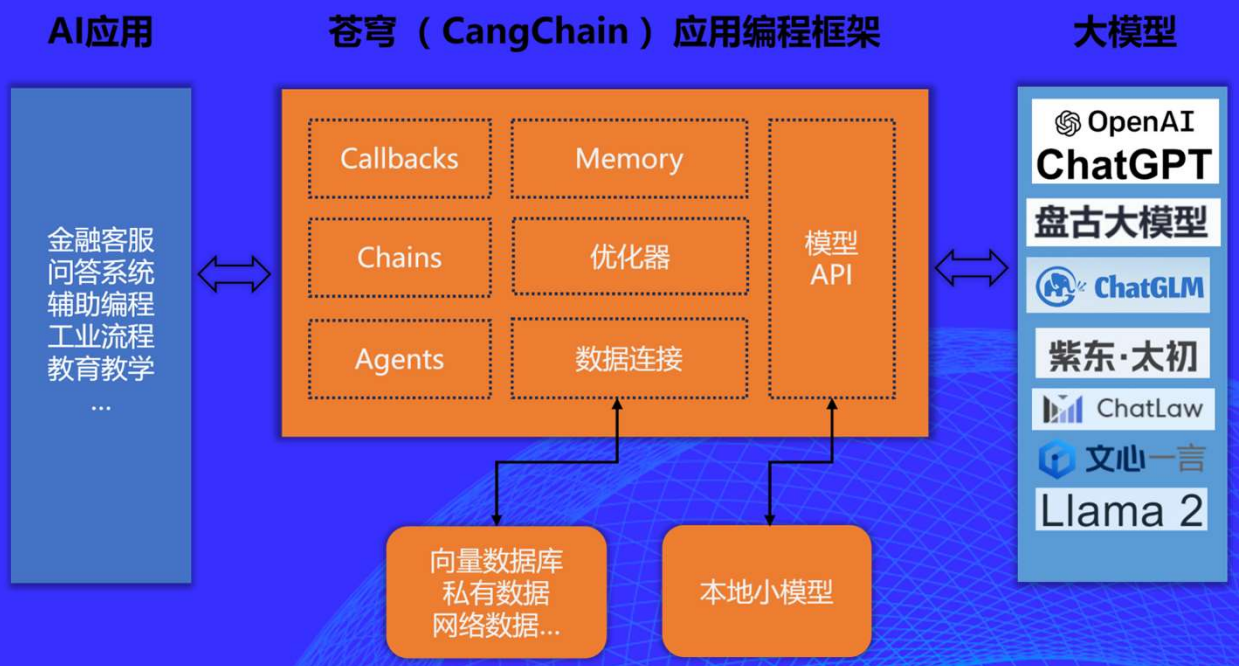
```
{
 " title " : " TOML Example "
 " owner " : {
   " name " : " Tom "
   " age " : 19
  }
}
```

仓颉与LLM编程框架

苍穹 CangChain

苍穹用来形容广阔的天空、壮阔的景象，代表着壮阔、辽阔的意境。比如《诗经》中的“苍苍者天”，《庄子》中的“苍苍乎如在其上”的描述。苍穹常常被用来比喻高远的理想或抱负，也可以指代神话中的天空之神。

苍穹(CangChain)框架服务于软件厂商、模型厂商，帮助终端用户快速开发AI应用



仓颉与LLM编程框架

苍穹结合仓颉的语言特点：

- 在苍穹模块实现，通过仓颉的类型系统，提升代码的安全性。类型安全、推断安全
- 苍穹多语言支持，基于仓颉语言的高效跨语言的特点，目标支持Python、Java、领域易扩展、高效构建领域抽象go、Javascript/TypeScript/ETS、Wasm；基于仓颉语言的领域易扩展的特点，目标支持一个内嵌领域语言eDSL。
  - 仓颉**跨语言互操作(**FFI) 是一种机制，通过该机制，一种编程语言写的程序可以调用另外一种编程语言编写的函数。
  - 仓颉语言的**元编程**是基于语法实现的。编译器在语法分析的阶段可以完成编写或操作目标程序的工作，用于操作目标程序的程序我们称为元程序。

CangJie

```
foreign func rand(): Int32
foreign func printf(fmt: CString, ...): Int32
main() {
    // call this function by `unsafe` block
    let r = unsafe { rand() }
    println("random number ${r}")
    unsafe {
        var fmt = LibC.mallocCString("Hello, No.%d\n")
        printf(fmt, 1)
        LibC.free(fmt)
    }
}
```

CangJie

```
from std import ast.*
main() {
    let tokens: Tokens = quote(1 + 2)
    // parseBinaryExpr is API provided by libast to parse input Tokens.
    // BinaryExpr is type provided by libast.
    var ast: BinaryExpr = parseBinaryExpr(tokens)
    let a = quote($ast) // without parentheses
    let b = quote($(ast)) // with parentheses
    let c = quote($ast.getLeftExpr()) // without parentheses
    let d = quote($(ast.getLeftExpr())) // with parentheses
    return 0
}
```

苍穹（CangChain）的进展

调研任务

任务	状态	时间	责任人
Semantic kernel 的调研	完成	2023.7	
Titokens 调研	完成	2023.7	
联邦学习的调研	完成	2023.7	
提示词优化、向量数据库的 client 和 sentence transformer 的调研	完成	2023.7	
Langchain 和 openAI 的 API 的一些调研	完成	2023.7	
多语言支持的调研	进行中	2023.8	

设计任务

任务	状态	时间	责任人
Agents 的设计	进行中	2023.7	
Chains 的设计	完成	2023.7	
Titokens 的设计	进行中	2023.7	
Chroma 向量数据库接口设计	进行中	2023.7	

实现任务

任务	状态	时间	责任人
LLM 的 API 封装	完成	2023.8	
Agents 实现	进行中	2023.8	
Titoken 库	未开始	2023.9	
向量数据库的 client 库	未开始	2023.9	
本地向量化库 sentence transformer	未开始	2023.9	
Onnx 库	未开始	2023.9	

Cangjie Community

仓颉语言KOL交流会

演示Demo

问题定义：在自然语言处理中，序列的词频分布、长度差异较大，模型难拟合。

Dataset Loader → Tokenizer → BoW model → Vectoring → Feature Extractor → FCM Cluster

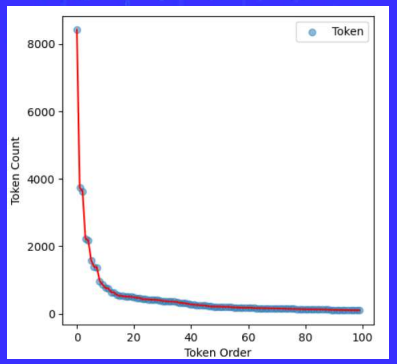
Sequence

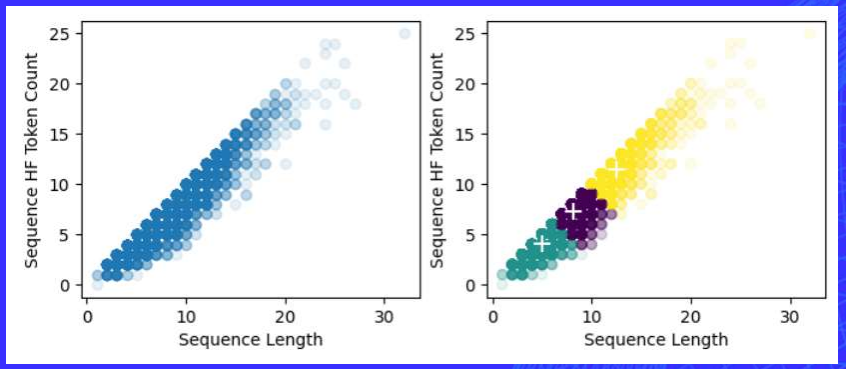
1. Go.
2. I left.
3. Tom has been behaving weirdly.
4. Oh, I'm sorry. I guess I have the wrong number.

Vector

1. [3]
2. [4,10]
3. [12,34,56,2,11]
4. [22,34,51,71,33,13,61,101,201,32]

Feature





Demo代码地址：https://gitee.com/chinesebear/kol

Cangjie Community
仓颉语言KOL交流会

Cangjie Community

仓颉语言 KOL 交流会

THANKS!

把握仓颉新机　蓄势引领辉煌

仓颉语言特性1

仓颉编程语言是**静态**的**强类型**语言，通过编译时检查尽早发现程序错误，排除运行时的错误。

CangJie

```
let myNumber: Int8 = 42                //整数类型
let name: String = "Rocky Balboa"      //字符串类型
let PI: Float64 = 3.141592             //浮点类型
// add() 函数包含两个整数类型的参数，返回值也是整数类型
func add(a: Int32, b: Int32): Int32 {
    return a + b
}

main(){
    println(myNumber)
    println(name)
    println(PI)
    println(add(3,22))
}
```

Python

```
myNumber = 42              #整数类型
name = "Rocky Balboa"      #字符串类型
PI = 3.141592              #浮点类型
# add() 包含两个参数和一个返回值
def add(a, b):
    return a + b


def main():
    print(myNumber)
    print(name)
    print(PI)
    print(add(3,22))
```

仓颉语言特性2

仓颉中的基本数据类型以及它们支持的基本操作，包括：整数类型、浮点类型、布尔类型、字符类型、字符串类型、 **Unit 类型**、元组类型、区间类型、 **Nothing 类型**

- **Unit 类型：**对于那些只关心副作用而不关心值的表达式，它们的类型是 Unit。
- **Nothing 类型：**一种特殊的类型，它不包含任何值，并且 Nothing 类型是所有类型的子类型。

注：目前编译器还不允许在使用类型的地方显式地使用 Nothing 类型

CangJie

```
func foo(i: Int64): Unit{
    println(i)
}

main() {
    for (i in 0..3){
        if (i == 0){
            continue //Nothing
        }
        if (i == 3){
            break // Nothing
        }
        foo(i) // Unit
    }
}
```

仓颉语言特性3

Collection 类型，包含 Array、 ArrayList、 HashSet、 HashMap。我们可以在不同的场景中选择适合我们业务的类型：

- Array：如果我们不需要增加和删除元素，但需要修改元素，就应该使用它。
- ArrayList：如果我们需要频繁对元素增删查改，就应该使用它。
- HashSet：如果我们希望每个元素都是唯一的，就应该使用它。
- HashMap：如果我们希望存储一系列的映射关系，就应该使用它。

Gitee社区有immutable_collection/collection-extension等collection的扩展库，主要包括中间操作、终结操作、转化操作、链表、栈、不可变列表、不可变映射、不可变栈.....

```
from std import collection.*
var c1: Array<Int64> = [1,2,3,4,5]
var c2: ArrayList<Int64> = ArrayList<Int64>([1,2,3,4,5])
var c3: HashSet<Int64> = HashSet<Int64>([1,2,3,4,5])
var c4: HashMap<String, Int64> = HashMap<String,Int64>([("1",1),("2",2),("3",3),("4",4),("5",4)])
```

CangJie