BM4KG 技术选型讨论

卢雨轩

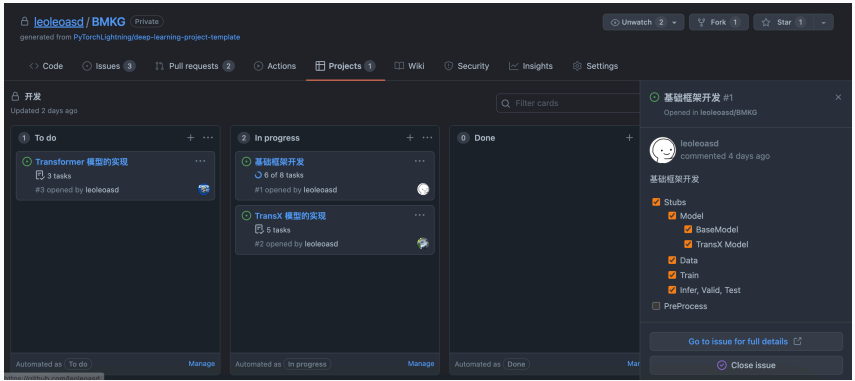
2022 年 3 月 29 日

主要内容

1. 目前进度
2. 技术选型

目前进度

目前进度



- 基础框架的 stubs 已经完成
- 正在开发数据预处理图分割部分

开发流程

- 每个人在自己的分支开发 (dev/xxxx)
- 通过 Pull Request 合并进入主分支
- 合并前 rebase master
  - `git fetch; git rebase origin/master`

技术选型

C++ or Rust?

技术栈对比:

C++	Rust
CPython	PyO3
numpy C-API	rust-numpy
OpenMP	Rayon
std::async	Tokio / std

技术选型

Why Rust?

Why Rust?

```
PyMODINIT_FUNC PyInit_spam(void) {
    PyObject *m;
    m = PyModule_Create(&spammodule);
    if (m == NULL) return NULL;
    SpamError = PyErr_NewException("spam.error", NULL, NULL);
    Py_XINCREF(SpamError);
    if (PyModule_AddObject(m, "error", SpamError) < 0) {
        Py_XDECREF(SpamError);
        Py_CLEAR(SpamError);
        Py_DECREF(m);
        return NULL;
    }
    return m;
}
```

Why rust? – Cleaner Interface

```rust
create_exception!(bmkg, MyError, pyo3::exceptions::PyException);
#[pymodule]
fn bmkg(_py: Python, m: &PyModule) -> PyResult<()> {
    m.add("MyError", _py.get_type::<MyError>())?;
    Ok(())
}
```

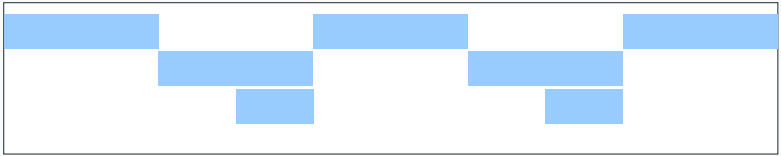
Why Rust? – Safer

```c
PyMODINIT_FUNC PyInit_spam(void) {
    PyObject *m;
    m = PyModule_Create(&spammodule);
    if (m == NULL) return NULL;
    SpamError = PyErr_NewException("spam.error", NULL, NULL);
    Py_XINCREF(SpamError);
    if (PyModule_AddObject(m, "error", SpamError) < 0) {
        Py_XDECREF(SpamError);
        Py_CLEAR(SpamError);
        Py_DECREF(m);
        return NULL;
    }
    return m;
}
```

Py_XINCREF? Py_XDECREF? Py_CLEAR? Py_DECREF?

Why Rust? – Easy Async Programming

Forward
Backward
Sampling



Why Rust? – Easy Async Programming

```rust
async fn heavy_task() -> PyResult<String> {
    println!("Doing heavy compute task...");
    // Sleep 2 seconds
    tokio::time::sleep(time::Duration::from_secs(2)).await;
    Ok(String::from("Hooray!"))
}
#[pyfunction]
fn negative_sampling_prepare(data: &PyDict, file_name: &str, partition:
↪  u32) -> PyResult<GraphPartitionResult> {
    let rt = Runtime::new()?;
    let handle = rt.spawn(heavy_task());
    Ok(GraphPartitionResult{
        rt,
        handle: Some(handle),
    })
}
```

Why Rust? – Easy Async Programming

```
future = negative_sampling_prepare(...)
some_heavy_compute_task()
result = future.wait()
```

技术选型

Some Concerns

Some Concerns – Is it stable?

- Is it stable?

Some Concerns – Is it stable?

- Is it stable?
- Yes!

Some Concerns – Is it stable?

- Is it stable?
- Yes!
- PyO3 **的第一个** Release **发布于** 2017 **年**
- **至今已有** 5 **年，文档、教程等资料非常丰富**

Some Concerns – Is it suitable for industrial usage?

- Is it suitable for industrial usage?

Some Concerns – Is it suitable for industrial usage?

- Is it suitable for industrial usage?
- Yes!

Some Concerns – Is it suitable for industrial usage?

- Is it suitable for industrial usage?
- Yes!
- **一个例子**:
  - Huggingface **的** Tokenizer **包（于** 2020 **年开始开发）就使用** PyO3

    > Extremely fast (both training and tokenization), thanks to the Rust implementation. Takes less than 20 seconds to tokenize a GB of text on a server's CPU.

Some Concerns – 环境好配吗？

- 环境好配吗？

Some Concerns – 环境好配吗？

- 环境好配吗？
- Yes!

Some Concerns – 环境好配吗？

- 环境好配吗？
- Yes!
- 开发过程中调试代码：
  - 安装 Rustup：`https://rustup.rs/` 一键安装
  - 安装 maturin：`pip install maturin`
  - 编译代码：`maturin develop -r`
  - 实验室服务器已经安装好
- 发布 pip 包：
  - 可直接编译二进制 wheel 发布
  - 源码发布需要用户下载安装 Rustup

Some Concerns – Maintainability ?

- 如何维护

Some Concerns – Maintainability ?

- 如何维护
- 我会持续维护
- 清华本科的优秀同学应该都会 Rust

Any Thoughts?