2 TensorFlow 环境搭建

TensorFlow 依赖的最主要的工具包是——**Protocol Buffer** 和 **Brazel**.

2.1 Protocol Buffer

Protocol Buffer 是谷歌开发的处理结构化数据的工具。何为结构化数据？举个例子，假设要记录一些用户信息，每个用户的信息包括用户的名字、**ID** 和 **E-mail** 地址。那么一个用户的信息可以表示成如下的形式：

---

name: 张三

id: 12345

email: zhangsan@abc.com

以上的用户信息就是一个结构化数据。其实也就是所谓的拥有多种属性的数据。要将结构化的用户信息持久化或进行网络传输时，就需要先将它们序列化。所谓序列化，是将结构化数据变成数据流的格式，简单地说就是变成一个字符串。将结构化的数据序列化，并从序列化之后的数据流中还原出原来的结构化数据，统称为处理结构化数据，这就是 **Protocol Buffer** 解决的主要问题。**Protocol Buffer** 序列化出来的数据要比 **XML** 格式的数据小 **3** 到 **10** 倍，解析时间要快 **20** 到 **100** 倍。

```
message user{
```

```
  optional string name = 1;
  required int32 id = 2;
}
```

这个里边 message 代表了一类结构化数据，比如这里的用户信息。Message 里面定义了每一个属性的类型和名字。Protocol Buffer 也定义了一个属性是必需的（required）而且是可选的（optional），或者是可重复的（repeated）。如果一个属性是可选的（optional），那么这个属性的取值可以为空；如果一个属性是可重复的（repeated），那么这个属性的取值可以是一个列表。举个例子：所有用户都需要有 ID，所以 ID 这个属性是必需的；不是所有用户都填写了姓名，所以姓名以这个属性是可选的；一个用户可能有多个 E-mail 地址，所以 E-mail 地址是可重复的。

2.2 Bazel

Bazel 是从谷歌开源的自动化创建工具，谷歌内部绝大部分的应用都是通过它来编译。

Workspace 是 Bazel 的一个基本概念。一个项目空间可以简单地理解为一个文件夹，在这里文件夹中包含了编译一个软件所需要的源代码以及输出编译结果的软链接地址。一个项目空间可以只包含一个应用比如 Tensor Flow.一个项目空间也可以包含多个应用。一个项目空间对应的文件夹是这个项目的根目录，在这个根目录中需要有一个 WORKSPACE 文

件，此文件定义了对外部资源的依赖关系。空文件同样也是一个合法的 WORKSPACE 文件。

Bazel 的编译方式是实现设定好的。因为 Tensor Flow 主要使用 Python 语言，Bazel 对 Python 支持的编译方式只有三种：py_binary、py_library 和 py_test。

我来解释一下:

py_binary 用于将 Python 程序编译为可执行文件

py_test 编译 Python 测试程序

py_library 将 Python 程序编译成库函数供其他 py_binary 或者是 py_test 调用。

举个例子吧：

hello_lib.py 完成打印“Hello World”的简单功能，它的代码如下：

```
def print_hello_world():
    print(“Hello World”)
```

hello_main.py 通过 hello_lib 通过调用 hello_lib.py 中定义的函数来完成输出，它的代码如下：

```
import hello_lib
hello_lib.print_hello_world()
```

在 BUILD 文件中定义了两个编译目标：

```
py_library(
    name = “hello_lib”,
    srcs = [
        “hello_lib.py”,
    ]
)
py_binary(
   name = “hello_main”,
   srcs = [
       “hello_main.py”,
     ],
     deps = [
           “:hello_lib,
           ],
)
```

BUILD 文件是由一系列编译目标组成的。定义编译目标的先后顺序不会影响编译结果。

在每一个编译目标第一行要给出编译方式，在这个例子里 Py_library 或者 py_binary。

Deps 给出了编译所需要的依赖关系，比如样例中的 hello_lib 作为依赖关系，比如样例中 hello_main.py 需要调用 hello_lib.py 中的函数，所以 hello_main 的编译目标中将 hello_lib 作为依赖关系。