# docker下tf\_serving多模型多版本部署

tensorflow的模型输出文件是冻结图pb文件,将模型文件通过tf\_serving部署成restful接口,采用docker下的tf\_serving,可以将宿主机的模型挂载到容器的指定目录下启动serving服务.分别测试**单模型单版本**,**多模型单版本**,**多模型多版本**,**单模型多版本**这四种情况下的部署方式.

#### 简单测试(单模型单版本部署)

docker拉取tf\_serving镜像

docker pull tensorflow/serving

tensorflow定义模型网络结构,定义模型的输入变量X和输出变量z

import tensorflow as tf

X = tf.placeholder("float") # 输入变量

Y = tf.placeholder("float")

W = tf.Variable(tf.random\_normal([1]), name="weight")

b = tf.Variable(tf.zeros([1]), name="bias")

z = tf.multiply(X, W) + b # 输出变量

tf.saved\_model.utils.build\_tensor\_info将X和z分别绑定为输入标签和输出标签到冻结图

from tensorflow.python.saved\_model import tag\_constants

builder = tf.saved\_model.builder.SavedModelBuilder(savedir + 'tfservingmodelv1')

inputs = {'input\_x': tf.saved\_model.utils.build\_tensor\_info(X)} # 定义输入签名，X为输入tensor

outputs = {'output': tf.saved\_model.utils.build\_tensor\_info(z)} # 定义输出签名， z为最终需要的输出结果tensor

signature = tf.saved\_model.signature\_def\_utils.build\_signature\_def(

inputs=inputs,

outputs=outputs,

method\_name=tf.saved\_model.signature\_constants.PREDICT\_METHOD\_NAME) # 指定任务类型是预测任务，预测任务包括分类，回归

builder.add\_meta\_graph\_and\_variables(sess, [tag\_constants.SERVING], {'my\_signature': signature})

builder.save()

生成模型冻结图文件结构如下

├──tfservingmodelv1

├── saved\_model.pb

└── variables

├── variables.data-00000-of-00001

└── variables.index

新建一个带有版本编号的文件夹001,将saved\_model.pb和variables移动到001下

├──tfservingmodelv1

├── 001

├── saved\_model.pb

└── variables

├── variables.data-00000-of-00001

└── variables.index

启动tf\_serving服务

tfservingmodelv1是宿主机模型文件目录  
models是容器下的模型目录,挂在到models下的linearregression目录下

docker run -t --rm -p 8501:8501 -v "/\*\*\*\*/\*\*\*\*/\*\*\*\*/\*\*\*\*/tfservingmodelv1:/models/linearregression/" -e MODEL\_NAME=linearregression tensorflow/serving

模拟URL请求

curl -d '{"instances": [1.0,2.0,5.0], "signature\_name":"my\_signature"}' -X POST http://localhost:8501/v1/models/linearregression:predict

{

"predictions": [1.92363, 3.89758, 9.81943

]

}

多个输入的情况,如果有多个input,需要传入json数组  
先定义多个输入标签

builder = tf.saved\_model.builder.SavedModelBuilder(pb\_path)

inputs = {'input\_x': tf.saved\_model.utils.build\_tensor\_info(lstm.input\_x),

'dropout\_keep\_prob': tf.saved\_model.utils.build\_tensor\_info(lstm.dropout\_keep\_prob)}

outputs = {'output': tf.saved\_model.utils.build\_tensor\_info(lstm.probs)}

signature = tf.saved\_model.signature\_def\_utils.build\_signature\_def(

inputs=inputs,

outputs=outputs,

method\_name=tf.saved\_model.signature\_constants.PREDICT\_METHOD\_NAME)

builder.add\_meta\_graph\_and\_variables(sess, [tag\_constants.SERVING], {'my\_signature': signature})

builder.save()

docker run -t --rm -p 8501:8501 -v "/\*\*\*\*/\*\*\*\*/\*\*\*\*/\*\*\*\*/tfservingmodel:/models/sentiment\_analysis/" -e MODEL\_NAME=sentiment\_analysis tensorflow/serving

curl -d '{"instances": [{"input\_x": [122, 91, 342, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], "dropout\_keep\_prob": 1}], "signature\_name":"my\_signature"}' -X POST http://localhost:8501/v1/models/sentiment\_analysis:predict

{

"predictions": [[0.594192, 0.405808]

]

}

#### docker tf\_serving的其他测试

（1）版本号命名：saved\_model.pb,variables必须放在一个带有版本标记的文件夹中，这个文件夹的命名必须全是数字，比如001，20200229，不能带有其他字符，标点，比如v1.01，1.01，2020-02-29  
（2）启动中模型文件删除替换：在docker镜像启动后，删除saved\_model.pb和variables不影响接口运行,如果替换了这两个文件，也不能热更新，必须更新文件之后，再重新启动docker的tensorflow\_model\_server服务才能更新预测结果  
（3）-v挂载运行，可以不加引号,-v用：间隔，分别是宿主机模型地址（pb文件目录地址）和挂载的地址，模型默认挂在/models/目录下，文件名随便取，但是要与启动服务中的MODEL\_NAME，以及请求体v1/models下的模型名称保持一致

docker run --rm -d -p 8501:8501 \

-v /Users/gengpeng/tensorflow\_project/churn\_lr.pb:/models/churn\_lr/ \

-e MODEL\_NAME=churn\_lr \

tensorflow/serving

其他参数：  
--rm：容器停止后就删除容器  
-d：后台运行  
-p：端口映射，宿主机端口:容器端口，REST的接口默认是8501，gRPC是8500，比如8502:8501是 正确的写法，8501:8502是错误的  
-v：挂载运行  
-e：设置环境变量  
（4）-mount挂载运行，这种可读性更好

docker run --rm -d -p 8501:8501 \

--mount type=bind,source=/Users/gengpeng/tensorflow\_project/churn\_lr.pb,target=/models/churn\_lr \

-e MODEL\_NAME=churn\_lr \

--name churn\_server \

tensorflow/serving

其他参数：  
--name：指定容器运行的NAMES，container\_id，可以直接用NAMES停止容器，docker container stop churn\_server  
type=bind：绑定挂载  
source：宿主机模型目录  
target：需要绑定挂载的地址  
（5）多版本号的优先级：  
001版本单独预测：0.512317657  
002版本单独预测：0.50290972  
两个版本移动到同一个pb文件目录下，结构如下：

在churn.pb文件下tree

├── 001

│ ├── saved\_model.pb

│ └── variables

│ ├── variables.data-00000-of-00001

│ └── variables.index

└── 002

├── saved\_model.pb

└── variables

├── variables.data-00000-of-00001

└── variables.index

docker run --rm -d -p 8501:8501 -v "/Users/gengpeng/tensorflow\_project/churn\_lr.pb:/models/churn\_lr/" -e MODEL\_NAME=churn\_lr tensorflow/serving

curl -d '{"instances": [{"input\_x": [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,0,0,0,1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0]}], "signature\_name":"my\_signature"}' -X POST http://localhost:8501/v1/models/churn\_lr:predict

{

"predictions": [0.50290972

]

**结论**：同一个模型，pb文件目录下如果存在多个模型版本，**会以最新版本为准，就是数字最大的**。

#### 多模型单版本部署

部署多模型不能指定MODEL\_NAME，需要一个配置文件**model.config**，多个MODEL\_NAME写在配置文件中了

tree multi\_model/

.

├── model.config

├── model1

│ └── 001

│ ├── saved\_model.pb

│ └── variables

│ ├── variables.data-00000-of-00001

│ └── variables.index

└── model2

└── 002

├── saved\_model.pb

└── variables

├── variables.data-00000-of-00001

└── variables.index

cat model.config

model\_config\_list:{

config:{

name: "model1",

base\_path: "/models/model1",

model\_platform: "tensorflow"

},

config:{

name: "model2",

base\_path: "/models/model2",

model\_platform: "tensorflow"

}

}

（1）name和Rest请求url中的模型名称:predict保持一致  
（2）base\_path是容器中的目录，默认是models下

启动docker多模型服务

docker run -d -p 8501:8501 \

--mount type=bind,source=/Users/gengpeng/tensorflow\_project/multi\_model/model1,target=/models/model1 \

--mount type=bind,source=/Users/gengpeng/tensorflow\_project/multi\_model/model2,target=/models/model2 \

--mount type=bind,source=/Users/gengpeng/tensorflow\_project/multi\_model/model.config,target=/models/model.config \

tensorflow/serving \

--model\_config\_file=/models/model.config

（1）将所有模型pb文件目录和配置文件，从宿主机挂载到容器的models下  
（2）挂载完成后，在tensorflow/serving后加入--model\_config\_file，指定容器中的配置文件路径

curl测试多模型

curl -d '{"instances": [{"input\_x": [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,0,0,0,1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0]}], "signature\_name":"my\_signature"}' -X POST

http://localhost:8501/v1/models/model1:predict

{

"predictions": [0.512317657

]

curl -d '{"instances": [{"input\_x": [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,0, 0, 0, 0, 0, 1 0, 0, 0, 0, 0, 1, 0, 0, 0,0,0,0,1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0]}], "signature\_name":"my\_signature"}' -X POST

http://localhost:8501/v1/models/model2:predict

{

"predictions": [0.50290972

]

#### 多模型多版本部署

model.config同级目录下有model1和model2两个模型,其中model2有两个版本001和002

tree

.

├── model.config

├── model1

│ └── 001

│ ├── saved\_model.pb

│ └── variables

│ ├── variables.data-00000-of-00001

│ └── variables.index

└── model2

├── 001

│ ├── saved\_model.pb

│ └── variables

│ ├── variables.data-00000-of-00001

│ └── variables.index

└── 002

├── saved\_model.pb

└── variables

├── variables.data-00000-of-00001

└── variables.index

cat model.config

model\_config\_list:{

config:{

name: "model1",

base\_path: "/models/model1",

model\_platform: "tensorflow"

},

config:{

name: "model2",

base\_path: "/models/model2",

model\_platform: "tensorflow",

model\_version\_policy:{

all:{}

}

}

}

curl -d '{"instances": [{"input\_x": [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,0, 0, 0, 0, 0, 1,0, 0, 0, 0, 0, 1, 0, 0, 0,0,0,0,1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0]}], "signature\_name":"my\_signature"}' -X POST

http://localhost:8501/v1/models/model2/versions/001:predict

{

"predictions": [0.486999243

]

curl -d '{"instances": [{"input\_x": [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,0, 0, 0, 0, 0, 1,0, 0, 0, 0, 0, 1, 0, 0, 0,0,0,0,1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0]}], "signature\_name":"my\_signature"}' -X POST

http://localhost:8501/v1/models/model2/versions/002:predict

{

"predictions": [0.50290972

]

#### 单模型的多版本部署

model.config同级目录下只有一个churn\_lr模型目录,该目录下有001,002两个版本的模型

multi\_model2

├── churn\_lr

│ ├── 001

│ │ ├── saved\_model.pb

│ │ └── variables

│ │ ├── variables.data-00000-of-00001

│ │ └── variables.index

│ └── 002

│ ├── saved\_model.pb

│ └── variables

│ ├── variables.data-00000-of-00001

│ └── variables.index

└── model.config

cat model.config

model\_config\_list:{

config:{

name: "churn\_lr",

base\_path: "/models/churn\_lr",

model\_platform: "tensorflow",

model\_version\_policy:{

all:{}

}

}

}

启动docker服务

docker run -d -p 8501:8501 \

--mount type=bind,source=/Users/gengpeng/tensorflow\_project/multi\_model2/churn\_lr,target=/models/churn\_lr \

--mount type=bind,source=/Users/gengpeng/tensorflow\_project/multi\_model2/model.config,target=/models/model.config \

tensorflow/serving \

--model\_config\_file=/models/model.config

接口测试

curl -d '{"instances": [{"input\_x": [0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,0, 0, 0, 0, 0, ,0, 0, 0, 0, 0, 1, 0, 0, 0,0,0,0,1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0]}], "signature\_name":"my\_signature"}' -X POST http://localhost:8501/v1/models/churn\_lr/versions/001:predict