Docker buildx 打包Ai-detect

配置buildx环境

1、启用 binfmt_misc

```
#使Docker for Linux 支持构建 arm 架构镜像
docker run --rm --privileged tonistiigi/binfmt:latest --install all
```

2、配置适用构建多平台镜像builder

```
#使Docker可支持同时指定多个 --platform
docker buildx create --use --name=mybuilder-cn --driver docker-container --
driver-opt image=dockerpracticesig/buildkit:master
```

```
docker buildx use mybuilder-cn
```

3、构建镜像

```
#打包镜像到远程仓库
docker buildx build --platform linux/arm64 -t
ccr.ccs.tencentyun.com/gizwits_gems/qcr4t-algorithm-arm64:test -f
build/Dockerfile --build-arg VERSION="test" . --push
#打包镜像到本地，使用docker load myimage.tar加载
docker buildx build --platform linux/arm64 -t
ccr.ccs.tencentyun.com/gizwits_gems/qcr4t-algorithm-arm64:test -o
type=docker,dest=- . > myimage.tar  -f build/Dockerfile --build-arg
VERSION="test" .
```

4、VIM3镜像运行结果

```
b'{"params": {"lambd_h": 14.0, "lambd_v": 13.7, "r_lambd_h": 15.6, "r_lambd_v":
13.8, "sigma_h": 4.5, "sigma_v": 4.5}, "result": {"weft": [0.446875,
0.25416666666666665, 0.4234375]}}'
36.09151077270508,

b'{"params": {"lambd_h": 14.0, "lambd_v": 13.7, "r_lambd_h": 15.6, "r_lambd_v":
13.8, "sigma_h": 4.5, "sigma_v": 4.5}, "result": {"weft": [0.384375, 0.3609375,
0.30885416666666665, 0.5583333333333333, 0.5135416666666667, 0.44166666666666665,
0.39114583333333336]}}'
2.412828207015991,
```

Jenkinds、WebHook构造镜像

Webhook配置

1、使用ngrok内网穿透，使得本机暴露出8080端口，使得外网可以触发jenkins

```
#下载ngrok
./ngrok http 8080
#得到url：http://13b4523f4dfb.ngrok.io
```

2、添加Webhook URL，设置Tag push events（pipelineTesT是Jenkins API Token ）

```
http://13b4523f4dfb.ngrok.io/generic-webhook-trigger/invoke?
token=pipelineTest&image_name=test
```

配置Jenkins

1、构造Jenkins镜像和运行

```
FROM jenkins/jenkins:lts
USER root

RUN apt-get update \
    && apt-get -y install \
    maven \
    nodejs \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg2 \
    software-properties-common \
    && curl -fsSL https://download.docker.com/linux/$(. /etc/os-release; echo
"$ID")/gpg | apt-key add - \
    && add-apt-repository \
      "deb [arch=amd64] https://download.docker.com/linux/$(. /etc/os-release;
echo "$ID")  $(lsb_release -cs) stable" \
    && apt-get update \
    && apt-get -y install docker-ce \
```

```
docker run -it -d --name myJenkins -p 8080:8087 -p 50000:50000 -v
/var/run/docker.sock:/var/run/docker.sock jenkins/jenkins:lts
```

2、完成Jenkins初始化（基本流程，下载Git Parameter、Generic Webhook Trigger插件、登陆个人dockerhub账号）

```
    #注意：镜像需要修改sh软连接为bash
    ls -al /bin/sh
    sudo ln -fs /bin/bash /bin/sh
```

3、配置ssh和Jenkins API Token

```
pipelineTest：" 1107fbbc7d31b0087aae51e6fc827da21d"
```

4、新建pipeLine，填入信息

```
1、构建触发器：使用：Generic Webhook Trigger
    Post content parameters填入：objectKind、gitHTTPURL等参数；
    Token：填入pipelineTest
2、流水线：使用Pipeline script from SCM。然后填入gitlab代码仓库：
git@gitlab.gizwits.com:test2/test.git。           并加载Jenkinsfile.groovy脚本（已在
gitlab）
3、添加凭证：使用ssh私钥验证(获取credentialsId)
```

5、修改2.groovy脚本中

```
credentialsId: '9c8aeb72-178e-41a1-8cae-db7e6fdf0db5',
#修改build仓库
 docker buildx build -f Dockerfile --platform 'linux/arm64'  -t
'2267024990/hello' . '--push'
```

6、构建JOb,在本机给仓库打上tag标签。Jenkins执行脚本，构建目标镜像。并成功push到dockerhub仓库