Milk-V Duo开发板实战——基于ShuffleNetV2的的图像分类

本教程介绍使用**TPU-MLIR**工具链对**ShuffleNetV2**的**PyTorch**模型进行转换，生成MLIR以及MLIR量化成INT8模型，并在**Milk-V Duo**开发板上进行部署测试，完成图像分类任务，涉及以下步骤：

> 🏳【注意】🏳：**Milk-V Duo开发板**搭载的是**CV1800B芯片**，该芯片支持**ONNX系列**和**Caffe模型**，目前不支持TFLite模型量化数据类型方面，目前支持**BF16格式的量化、INT8格式的非对称量化**

1. 工作环境准备
2. ShuffleNetV2-PyTorch模型转换
3. 部署 INT8 cvimodel 到Duo开发板并验证

以下对此3个步骤展开详细介绍。

1. 工作环境准备

1.1 配置docker开发环境

安装并配置docker:

```
sudo apt install docker.io
sudo systemctl start docker
sudo systemctl enable docker
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
```

从docker hub拉取镜像文件：

```
docker pull sophgo/tpuc_dev:v2.2
```

运行docker创建容器，其中的`duodev`是容器名称，可自行修改；创建后默认目录为`/workspace`：

```
docker run --privileged --network=host --name duodev -v $PWD:/workspace -
it sophgo/tpuc_dev:v2.2
```

docker环境内配置网络并安装基本依赖：

```
apt-get update
apt-get install net-tools
```

下载tpu-mlir模型转换工具链，包命名格式为`tpu-mlir_xxxx.tar.gz`，其中`xxxx`为版本号，此教程以版本`v1.2.89-g77a2268f-20230703`为例：

```
wget --user='cvitek_mlir_2023' --password='7&2Wd%cu5k'
ftp://218.17.249.213/home/tpu-mlir_v1.2.89-g77a2268f-20230703.tar.gz
```

解压工具链并导入环境变量：

```
tar zxf tpu-mlir_v1.2.89-g77a2268f-20230703.tar.gz
source tpu-mlir_v1.2.89-g77a2268f-20230703/envsetup.sh
```

1.2 准备工作目录

创建并进入 `shufflenet_v2` 目录，将`tpu-mlir` 工具链目录（后文用 `${TPUMLIR_DIR}` 指代）下的图片文件放入此目录下：

```
# 创建并进入目录
mkdir shufflenet_v2 && cd shufflenet_v2

# 拷贝测试图片
cp -rf ${TPUMLIR_DIR}/regression/dataset/ILSVRC2012/ .
cp -rf ${TPUMLIR_DIR}/regression/image/ .
```

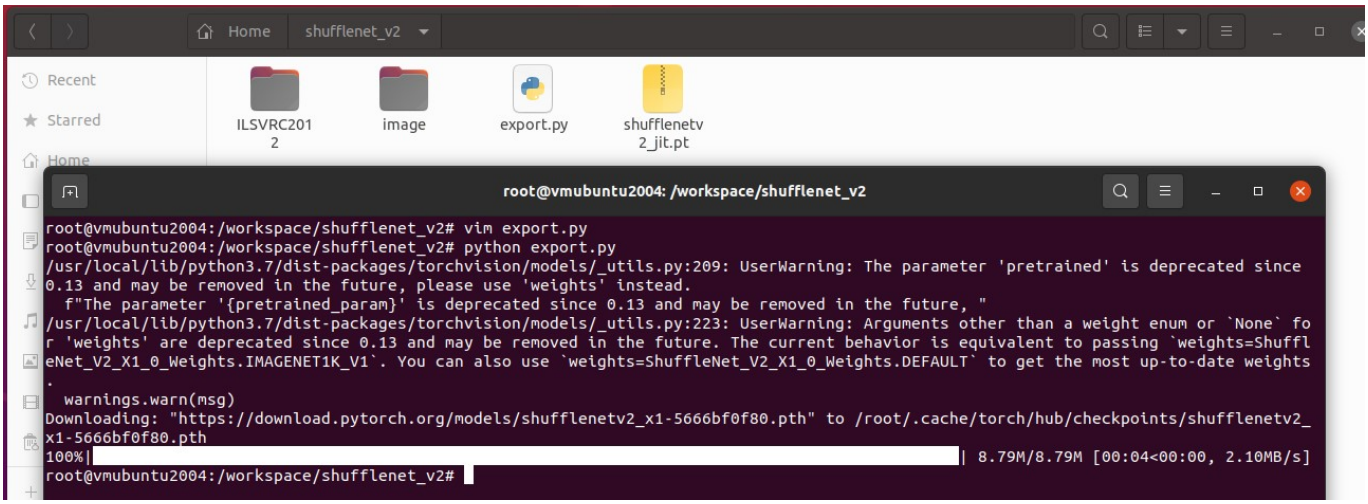
创建名为 `export.py` 文件，并在文件中写入如下代码：

```
import torch
from torchvision.models.shufflenetv2 import shufflenet_v2_x1_0

model = shufflenet_v2_x1_0(pretrained=True)
model.eval()
torch.jit.trace(model, torch.randn(1, 3, 640,
640)).save("./shufflenetv2_jit.pt")
```

运行 `export.py` 文件：

```
python export.py
```



创建名为workspace的工作目录（后文用 ${WORK_DIR} 指代），用于存放编译生成的MLIR、cvimodel等文件：

```
mkdir workspace && cd workspace
```

2. ShuffleNetV2-PyTorch模型转换

模型转换步骤如下：

- PyTorch模型转换成MLIR
- 生成量化需要的校准表
- MLIR量化成 INT8 非对称cvimodel

2.1 PyTorch模型转换成MLIR

模型输入是图片, 在转模型之前我们需要了解模型的预处理。如果模型用预处理后的npz文件做输入, 则不需要考虑预处理。预处理过程用公式表达如下(x代表输入): $$ y = (x-mean)\times scale $$

本例中的模型是 **BGR** 输入, mean和scale分别为 103.94,116.78,123.68 和 0.017,0.017,0.017，模型转换命令如下:

```
model_transform.py \
    --model_name shufflenet_v2 \
    --model_def ../shufflenetv2_jit.pt \
    --input_shapes [[1,3,224,224]] \
    --resize_dims=256,256 \
    --mean 103.94,116.78,123.68 \
    --scale 0.017,0.017,0.017 \
    --pixel_format bgr \
    --test_input ../image/cat.jpg \
    --test_result shufflenet_v2_top_outputs.npz \
    --mlir shufflenet_v2.mlir
```

执行 model_transform.py 脚本生成的文件如下图所示：

```
min_similiarity = (0.9999997615814209, 0.999997907542016, 113.3873176574707)
Target    shufflenet_v2_top_outputs.npz
Reference shufflenet_v2_ref_outputs.npz
npz compare PASSED.
compare 1166: 100%|                                                                                | 180/180 [00:05<00:00, 30.29it/s]
[Success]: npz_tool.py compare shufflenet_v2_top_outputs.npz shufflenet_v2_ref_outputs.npz --tolerance 0.99,0.99 --except - -vv
root@vmubuntu2004:/workspace/shufflenet_v2/workspace# ls
shufflenet_v2.mlir        shufflenet_v2_origin.mlir                shufflenet_v2_top_f32_all_weight.npz
shufflenet_v2_in_f32.npz  shufflenet_v2_top_f32_all_origin_weight.npz  shufflenet_v2_top_outputs.npz
root@vmubuntu2004:/workspace/shufflenet_v2/workspace#
```

2.2 生成量化需要的校准表

运行 `run_calibration.py` 得到校准表，输入数据的数量根据情况准备100~1000张左右。 这里用现有的100张来自ILSVRC2012的图片举例，执行calibration命令：

```
run_calibration.py shufflenet_v2.mlir \
    --dataset ../ILSVRC2012 \
    --input_num 100 \
    -o shufflenet_v2_cali_table
```

运行完成后会生成名为 `${model_name}_cali_table` 的文件, 该文件用于后续编译INT8模型的输入文件。

```
activation_collect_and_calc_th for op: 1166: 100%|                                | 206/206 [01:20<00:00,  2.55it/s]
[2048] threshold: 1166: 100%|                                                     | 206/206 [00:00<00:00, 430.30it/s]
prepare data from 100
tune op: 1166: 100%|                                                              | 206/206 [01:57<00:00,  1.75it/s]
auto tune end, run time:118.10031175613403
root@vmubuntu2004:/workspace/shufflenet_v2/workspace# ls
shufflenet_v2.mlir        shufflenet_v2_in_f32.npz    shufflenet_v2_top_f32_all_origin_weight.npz  shufflenet_v2_top_outputs.npz
shufflenet_v2_cali_table  shufflenet_v2_origin.mlir   shufflenet_v2_top_f32_all_weight.npz
root@vmubuntu2004:/workspace/shufflenet_v2/workspace#
```

2.3 MLIR量化成 INT8 非对称cvimodel

> 🏳【注意】🏳：**Milk-V Duo开发板**搭载的是**CV1800B芯片**，该芯片支持**ONNX系列**和**Caffe模型**,目前不支持TFLite模型量化数据类型方面，目前支持**BF16格式的量化、INT8格式的非对称量化**，故此节中使用`model_deploy.py`脚本参数使用**asymmetric**进行**非对称量化** 将MLIR文件转成INT8非对称量化模型，执行如下命令：

```
model_deploy.py \
    --mlir shufflenet_v2.mlir \
    --asymmetric \
    --calibration_table shufflenet_v2_cali_table \
    --fuse_preprocess \
    --customization_format BGR_PLANAR \
    --chip cv180x \
    --quantize INT8 \
    --test_input ../image/cat.jpg \
    --tolerance 0.96,0.72 \
    --model shufflenet_v2_cv1800_int8_asym.cvimodel
```

编译完成后, 会生成名为 `${model_name}_cv1800_int8_asym.cvimodel` 的文件，如下图所示：

```
min_similiarity = (1.0, 1.0, inf)
Target    shufflenet_v2_cv180x_int8_asym_model_outputs.npz
Reference shufflenet_v2_cv180x_int8_asym_tpu_outputs.npz
npz compare PASSED.
compare 1166: 100%|                                                                                | 2/2 [00:00<00:00, 30.05it/s]
[Success]: npz_tool.py compare shufflenet_v2_cv180x_int8_asym_model_outputs.npz shufflenet_v2_cv180x_int8_asym_tpu_outputs.npz --tole
rance 0.99,0.90 --except - -vv
root@vmubuntu2004:/workspace/shufflenet_v2/workspace# ls
_weight_map.csv                          shufflenet_v2_origin.mlir
shufflenet_v2.mlir                       shufflenet_v2_top_f32_all_origin_weight.npz
shufflenet_v2_cali_table                 shufflenet_v2_top_f32_all_weight.npz
shufflenet_v2_cv1800_int8_asym.cvimodel  shufflenet_v2_top_outputs.npz
shufflenet_v2_cv180x_int8_asym_final.mlir  shufflenet_v2_tpu_addressed_cv180x_int8_sym_weight.npz
shufflenet_v2_cv180x_int8_asym_tpu.mlir  shufflenet_v2_tpu_addressed_cv180x_int8_sym_weight_fix.npz
shufflenet_v2_in_ori.npz
root@vmubuntu2004:/workspace/shufflenet_v2/workspace#
```

3. 部署 INT8 cvimodel 到Duo开发板并验证

> 此文档不赘述Duo的工作环境配置，默认已能成功连接开发板，备注Duo开发板连接信息如下：
>
> IP: 192.168.42.1 user: root password: milkv

下载开发板上运行需要的`cvitek_tpu_sdk`：

```
wget --user='cvitek_mlir_2023' --password='7&2Wd%cu5k'
ftp://218.17.249.213/home/tpu_sdk_t4.1.0-14-
g3e77050/cvitek_tpu_sdk_cv180x_musl_riscv64_rvv.tar.gz
```

将该`cvitek_tpu_sdk`包上传到Duo开发板上：

```
scp cvitek_tpu_sdk_cv180x_musl_riscv64_rvv.tar.gz
root@192.168.42.1:/cvitek_tpu_sdk_cv180x_musl_riscv64_rvv.tar.gz
```

在`${WORK_DIR}`目录下，复制生成的`${model_name}_cv1800_int8_asym.cvimodel`到Duo开发板上：

```
scp shufflenet_v2_cv1800_int8_asym.cvimodel
root@192.168.42.1:/shufflenet_v2_cv1800_int_asym.cvimodel
```

注意：此节以下内容在**Duo开发板**上进行

---

ssh连接Duo开发板，可以看到刚才传输的`cvitek_tpu_sdk`包和`cvimodel`：

```
ssh root@192.168.42.1
```

```
[root@milkv]/# ls
bin                                     lib64                   proc                    tmp
cvitek_tpu_sdk                          linuxrc                 root                    usr
cvitek_tpu_sdk_cv180x_musl_riscv64_rvv.tar.gz  lost+found       run                     var
dev                                     media                   sbin
etc                                     mnt                     shufflenet_v2_cv1800_int_asym.cvimodel
lib                                     opt                     sys
[root@milkv]/#
```

解压cvitek_tpu_sdk_cv180x_musl_riscv64_rvv.tar.gz，导入环境变量，进入samples目录进行测试：

```
# 解压包
tar zxf cvitek_tpu_sdk_cv180x_musl_riscv64_rvv.tar.gz

# 导入cvitek_tpu_sdk的目录，例如本例中是TPU_ROOT=/cvitek_tpu_sdk
export TPU_ROOT=$PWD/cvitek_tpu_sdk

# 进入sdk目录并导入环境变量
cd cvitek_tpu_sdk && source ./envs_tpu_sdk.sh

# 打印cvimodel info，$MODEL_PATH为放cvimodel的目录
cd samples
./bin/cvi_sample_model_info
$MODEL_PATH/shufflenet_v2_cv1800_int_asym.cvimodel

# 测试
./bin/cvi_sample_classifier_fused_preprocess \
    $MODEL_PATH/shufflenet_v2_cv1800_int_asym.cvimodel \
    ./data/cat.jpg \
    ./data/synset_words.txt
```

```
[root@milkv]/cvitek_tpu_sdk/samples# ./bin/cvi_sample_classifier_fused_preprocess \
>         /shufflenet_v2_cv1800_int_asym.cvimodel \
>         ./data/cat.jpg \
>         ./data/synset_words.txt
version: 1.4.0
shufflenet_v2 Build at 2023-07-18 17:42:37 For platform cv180x
Max SharedMem size:301056
CVI_NN_RegisterModel succeeded
CVI_NN_Forward succeeded
------
  7.812500, idx 285, n02124075 Egyptian cat
  6.937500, idx 278, n02119789 kit fox, Vulpes macrotis
  6.781250, idx 331, n02326432 hare
  6.656250, idx 282, n02123159 tiger cat
  6.500000, idx 17, n01580077 jay
------
CVI_NN_CleanupModel succeeded
[root@milkv]/cvitek_tpu_sdk/samples#
```

注意：

1. sample目录下的samples_extra提供了更多samples脚本，但其中cvimodel名字已经硬编码在其中，如想使用脚本运行，需要自行修改cvimodel名字。
2. 此小节介绍的是使用预编译好的sample程序对转换好的cvimodel进行部署测试，如果开发者有兴趣对samples源码进行编码和交叉编译，请参考官网TPU-MLIR文档中的**第9章**《CV18xx芯片使用指南》中的**第3小节**"编译和运行runtime sample"内容。

附录

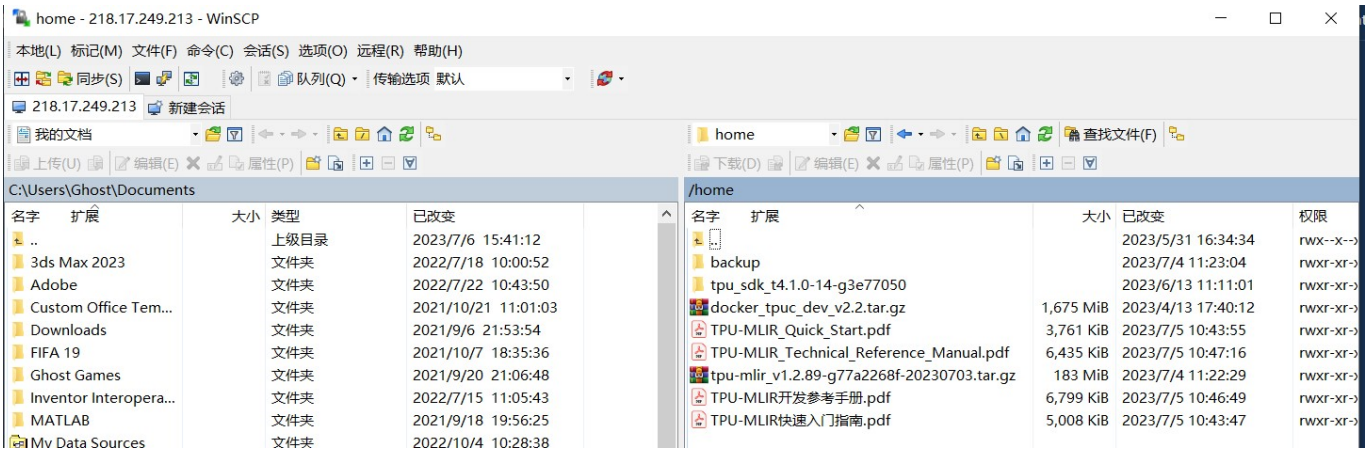
正文涉及到的文件总结如下：

- TPU-MLIR模型转换工具链：tpu-mlir_v1.2.89-g77a2268f-20230703.tar.gz

- TPU SDK开发工具包：cvitek_tpu_sdk_cv180x_musl_riscv64_rvv.tar.gz
- （附）Sample测试例程源码：cvitek_tpu_samples.tar.gz
- （附）转换好的cvimodel包：cvimodel_samples_cv180x.tar.gz

正文提到的TPU开发所需的包文件可在下面sftp站点获取：

> sftp://218.17.249.213 user: cvitek_mlir_2023 password: 7&2Wd%cu5k



或者直接使用wget获取：

```
# TPU-MLIR模型转换工具链
wget --user='cvitek_mlir_2023' --password='7&2Wd%cu5k'
ftp://218.17.249.213/home/tpu-mlir_v1.2.89-g77a2268f-20230703.tar.gz

# TPU SDK开发工具包
wget --user='cvitek_mlir_2023' --password='7&2Wd%cu5k'
ftp://218.17.249.213/home/tpu_sdk_t4.1.0-14-
g3e77050/cvitek_tpu_sdk_cv180x_musl_riscv64_rvv.tar.gz

#  （附）Sample测试例程源码
wget --user='cvitek_mlir_2023' --password='7&2Wd%cu5k'
ftp://218.17.249.213/home/tpu_sdk_t4.1.0-14-
g3e77050/cvitek_tpu_samples.tar.gz

#  （附）转换好的cvimodel包
wget --user='cvitek_mlir_2023' --password='7&2Wd%cu5k'
ftp://218.17.249.213/home/tpu_sdk_t4.1.0-14-
g3e77050/cvimodel_samples_cv180x.tar.gz
```