分类网络示例使用说明

说明

本示例支持两种推理方式：一种是读取一张本地图片，调用模型进行推理，并输出结果；另一种是通过读取摄像头数据进行推理，并通过DP接口将结果输出到显示器上。

考虑到简单通用性，该示例，从json文件中读取模型和图片信息，加载并执行。 执行时需要指定 相应的配置文件。 如

```
./image_classify ../configs/resnet50/drink.json
```

工程目录结构:

```
├── CMakeLists.txt // cmake 工程配置文件。
├── include
|   └── io // paddle_mobile 头文件目录
|       ├── paddle_inference_api.h
|       ├── type_define.h
|       └── types.h
├── configs // 配置文件目录
|   ├── Inceptionv2
|   |   └ zebra.json 斑马图片的配置
|   ├── Inceptionv3
|   |   └ zebra.json 斑马图片的配置
|   ├── mobilenetv1
|   |   └ zebra.json 斑马图片的配置
|   └── resnet50
|       └ drink.json 怡宝图片的配置
├── lib
|   └── libpaddle-mobile.so
├── models // 模型文件目录
|   ├── Inceptionv2
|   ├── Inceptionv3
|   ├── mobilenetv1
|   └── resnet50
|── src
|   ├── json.hpp // json 解析库
|   ├── video_classify.cpp // 视频推理示例
|   └── image_classify.cpp // 图片推理示例
└── README.md
```

下面是配置文件示例。

```
{
    "model":"../models/resnet50",
    "combined_model":true, //
    "input_width":224,
    "input_height":224,
    "image":"../models/resnet50/1.jpg",
    "mean":[104,117,124],
    "scale":1,
    "format":"BGR"
}
```

key	value
model	模型目录存放的位置
combined_model	是否为融合的模型，只有两个文件的是融合模型
input_width	输入网络的图片尺寸 输入图片会缩放到该大小
input_height	输入网络的图片尺寸
image	进行分类的图片输入
mean	平均值
scale	输入网络前预算处理为 (x - mean) * scale
format	网络所需要的格式，OpenCV默认是BGR

其它的分类网络也可以通过添加/修改 配置文件实现，无须修改代码。

使用步骤

1.加载驱动,系统启动后加载一次即可（默认已加载）

```
insmod /home/root/workspace/driver/fpgadrv.ko

//如果需要卸载驱动,则使用下面的命令
rmmod fpgadrv
```

2.编译示例，本设备具有编译能力，进入到sample/classification示例的build目录下进行编译

```
cd /home/root/workspace/sample/classification
// 如果没有build目录，创建一个
mkdir build
cd build
rm -rf *
// 调用cmake 创建 Makefile
cmake ..
// 编译工程。
make
```

编译结束后会在build 目录生成如下几个文件。 image_classify 读取本地图片推理示例。 video_classify 读取摄像头数据进行推理，要连接USB Camera才能使用。如需显示结果还要连接DP显示器或者HDMI显示器或者VGA显示器等。

3.执行示例

```
读取一张图片，做推理，将结果保存为图片输出result.jpg
./image_classify ../configs/resnet50/drink.json

读取摄像头数据，做推理，将结果通过DP接口显示到桌面显示屏上
./video_classify ../configs/resnet50/drink.json
```