浙江大學

《人工智能与大数据综合实践》

题目：　　基于华为 **MindSpore Lite** 的猫狗分类

姓名：　　朱语 沈骏一 宋孟炫

指导教师：　　刘勇、张建明、徐巍华

专业：　　自动化（控制）、自动化（电气）

所在学院：　　控制科学与工程学院、电气工程学院

日期：　　2023 年 7 月

基于华为 MindSpore Lite 的猫狗分类

朱语[1]、沈骏一[2]、宋孟炫[2]

[1]浙江大学电气学院、[2]浙江大学控制科学与工程学院

摘要

本实验报告旨在探索将复杂的神经网络模型应用于终端设备，实现智慧生活。基于华为 MindSpore 深度学习框架，mobilenet 网络构建了猫狗分类模型，并利用 MindSpore Lite 框架将其部署到手机上，实现模型的端侧应用。实验目标在于开发一款移动应用，用户通过拍摄或选择照片即可快速准确分类猫狗。实验通过 MindSpore 框架构建模型，使开发者掌握其使用方法，受益于其他领域的应用。在此基础上我们尝试了拓展实验数据、拓展分类模型网络等创新性实验，检验了分类模型在不同数据量、数据种类的数据集上展现的分类能力，以及不同模型对于相同数据集的分类所展现的识别能力提升。实验帮助我们掌握深度学习分类项目的开发流程，并尝试在其他目标任务中应用。浙江大学控制科学与工程学院与华为公司为本实验搭建了便捷好用的操作平台，并提供的充足丰富的实验指导。

关键词

深度学习、人工智能、神经网络、MindSpore、MobileNet、MindCV

Dog and cat classification based on Huawei MindSpore Lite

Yu ZHU[1]、Junyi SHEN[2]、Mengxuan SONG[2]

[1]College of Electrical Engineering, Zhejiang University、

[2]College of Control Science and Engineering, Zhejiang University

Abstract

The aim of this experiment report is to explore the application of complex neural network models on terminal devices to achieve smart living. Based on the Huawei MindSpore deep learning framework, a cat-dog classification model was constructed using the MobileNet network. The model was then deployed on mobile phones using the MindSpore Lite framework for edge-side applications. The objective of the experiment was to develop a mobile application that allows users to quickly and accurately classify cats and dogs by taking photos or selecting images.The experiment utilized the MindSpore framework to build the model, enabling developers to grasp its usage and apply it in other fields. Additionally, innovative experiments were conducted to expand the dataset and classification model network. These experiments examined the model's classification capability on datasets of varying sizes and types and evaluated the recognition performance improvements of different models on the same dataset.The experiment helped participants understand the development process of deep learning classification projects and encouraged exploration of applying similar techniques to other tasks. The College of Control Science and Engineering at Zhejiang University, in collaboration with Huawei, provided a user-friendly operational platform and extensive experimental guidance for this study.

Key Words

Deep learning; Artificial Intelligence; Neural Network; MindSpore; MindCV; MobileNet

Table of Contents

一、引言

在深度学习算法飞速发展的今天，越来越多的研究者希望复杂的神经网络模型可以应用于更多的场景，如应用于终端设备以实现智慧生活，本实验基于 MindSpore 深度学习框架构建猫狗分类模型，然后利用 mindspore lite 框架将模型部署到手机上，实现模型的端侧应用。为此，我们提出我们的实验选题：**“基于华为 MindSpore Lite 的猫狗分类”**，具体来说，是要通过深度学习算法和华为云平台，实现以下目标：

1. 通过 MindSpore 深度学习框架构建猫狗分类模型

2. 利用 mindspore lite 框架将模型部署到手机上，实现模型应用。

项目的意义与目的

该实验的开展对深度学习方向本科生的学习实践具有显著意义，具体如下所示：

1. 该项目的最初目标旨在开发一个移动应用程序，让用户能够通过拍摄或选择照片，快速准确地对图像中的猫狗进行分类。这样的应用可以为用户提供便捷的服务，例如帮助用户鉴别宠物的品种、帮助寻找失踪的宠物等。

2. 通过该项目，可以展示如何使用华为 MindSpore 框架构建一个猫狗分类模型。MindSpore 是一个面向端到端深度学习的开源框架，具有高效、易用、灵活等特点。该项目可以帮助开发人员了解和掌握 MindSpore 框架的使用方法，从而在其他领域的应用中受益。

3. 将深度学习模型部署到手机上是一项具有挑战性的任务。在这个项目中，通过使用 MindSpore Lite 框架，可以将训练好的猫狗分类模型进行轻量化处理，使其适应手机等资源受限的环境。这样一来，用户可以直接在手机上运行该模型，而无需依赖于云端计算，提供更快速的分类结果。

4. 通过对以上示例项目的开发，系统掌握类似基于深度学习的多目标/单目标分类项目的开发流程，并尝试更换数据集、模型架构便于进行其他目标任务的完成。

同时，浙江大学控制科学与工程学院与华为公司为本实验搭建了便捷好用的操作平台，并提供的充足丰富的实验指导。

实验指导

实验材料可以通过以下链接下载，下载后参考实验指导书的“深度学习”文件夹。

链接：https://pan.baidu.com/s/1-C3AIvAsGFffgUbvH_Uihw?pwd=1111

提取码：1111

二、正文

问题描述

猫狗分类问题是典型的基于图片数据集的多分类问题，其核心是利用 CNN 卷积神经网络，提取图片的关键特征，并形成对应的特征向量便于分类器进行分类，涉及到的核心难点如下所示：

- 模型适配问题

一般来说，我们在模型训练时采用的是市面上流行的 Pytorch 或者 Tensorflow 机器学习框架，因此需要解决将其他框架下预训练过的模型迁移到 mindspore lite 框架中的适配性问题。当使用其他深度学习框架进行训练时，模型的结构和参数可能与 MindSpore Lite 框架的要求不完全一致。因此，需要进行模型的适配和转换，以使其能够在 MindSpore Lite 框架中进行部署和推理。

- 模型性能提升

需要通过模型和参数的优化尽可能提升猫狗分类的准确率。我们预计采用如下方法：

a. 数据增强：通过对训练数据进行增强，如随机裁剪、旋转、翻转、缩放等，可以扩充训练集的多样性，帮助模型更好地学习特征并提高鲁棒性。

b. 模型架构优化：考虑优化模型的架构，如增加或减少网络层、调整层的尺寸、调整激活函数等，以提升模型的表示能力和性能。

c. 参数调优：通过调整模型的超参数，如学习率、权重衰减、批量大小等，可以提高模型的训练效果和泛化能力。

d. 损失函数选择：根据具体问题的特点，选择合适的损失函数，以便更好地衡量模型的性能和优化目标。

e. 模型集成：采用模型集成的方法，如投票、平均、堆叠等，将多个模型的预测结果综合起来，以提升分类的准确率和鲁棒性。

f. 迁移学习：利用预训练的模型，如在大规模图像数据集上训练好的模型，可以通过迁移学习将其特征提取能力转移到猫狗分类任务中，从而加速模型的训练和提升性能。

g. 模型蒸馏：通过使用较大的模型生成的软标签来辅助训练较小的模型，可以提高小模型的泛化能力和性能。

h. 自动化超参数调优：使用自动化的超参数调优方法，如网格搜索、贝叶斯优化等，可以自动搜索最佳的超参数配置，以提升模型性能。

- 模型瘦身

由于模型需要搭载到内存有限、计算能力有限的手机端，需要限制模型大小和模型的计算开销。一般有如下方法：

a. 参数量削减：通过减少模型中的参数数量来减小模型的大小。可以使用剪枝技术，识别和删除模型中冗余的连接和参数，以达到参数量减少的效果。还可以使用低秩近似方法，将参数矩阵替换为较小的矩阵，从而减少参数的数量。

b. 网络结构简化：通过简化模型的网络结构，如减少网络层数、减少卷积核的数量等，以减小模型的复杂度和计算开销。可以通过剪枝、模型压缩等技术来实现网络结构的简化。

c. 量化：将模型的权重和激活值从浮点数转换为较低精度的定点数或浮点数表示，从而减小模型的大小和计算开销。量化技术可以将模型参数和中间结果表示为更小的数据类型，如 8 位整数或浮点数。

d. 分解和逼近技术：使用矩阵分解、逐层逼近等技术将复杂的层拆分为多个简单的层，从而减小模型的规模和计算量。这些技术可以将大型卷积层转化为更小的卷积层或全连接层，减少计算和存储开销。

e. 模型蒸馏：通过使用较大的模型生成的软标签来辅助训练较小的模型，从而减小模型的规模。模型蒸馏通过传递更多的知识和信息，将大模型的性能转移到小模型中，以减少模型的大小和计算开销。

- 实时性要求

在移动设备上进行猫狗分类时，用户通常希望能够实时得到分类结果，以便及时采取相应的行动。该项目将通过在手机上部署轻量化的猫狗分类模型，以实现快速的分类结果输出，满足用户对实时性的需求。

解决方案

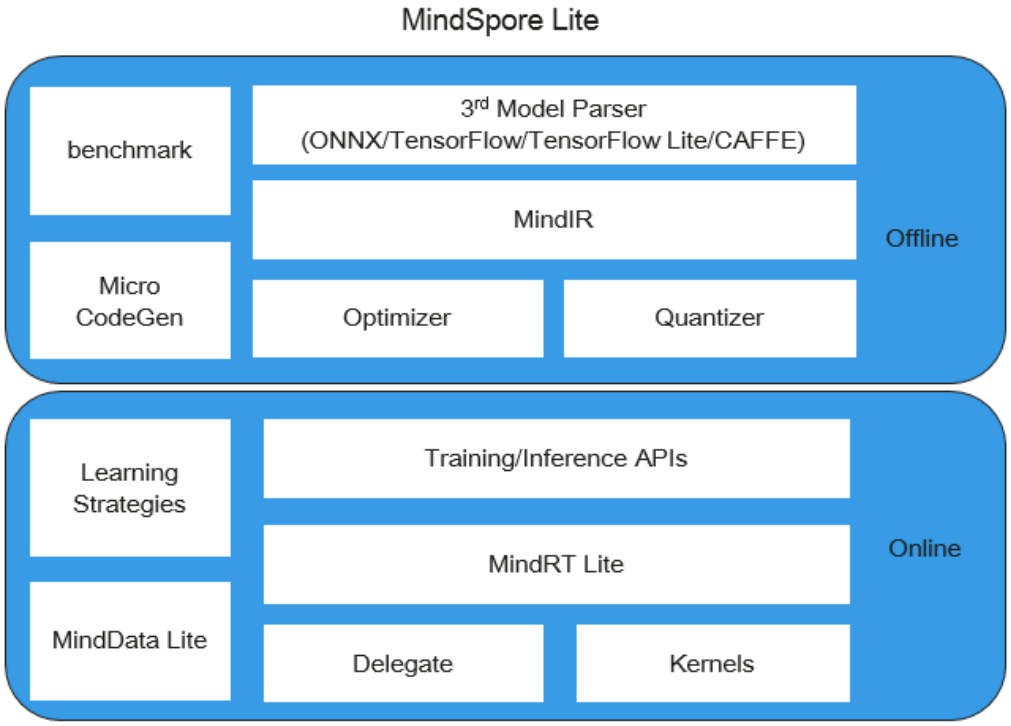
实验总体设计

本实验主要分为两部分，其中第一部分是基于 MindSpore 框架，通过 Fine-Tuning 训练一个猫狗分类模型，并部署到手机端；第二部分是解析 APP 构建代码，从而学习模型部署的执行逻辑。

mindspore lite 框架介绍

MindSpore Lite 是一个极速、极智、极简的 AI 引擎，使能全场景智能应用，为用户提供端到端的解决方案，帮助用户使能 AI 能力。

MindSpore Lite 支持 CPU/GPU/NPU，并且支持 IOS、Android 以及 LiteOS 等嵌入式操作系统，在模型方面支持 MindSpore/TensorFlow Lite/Caffe/Onnx 模型，极致性能、轻量化、全场景支持且高效部署。



mobilenet v2 介绍

MobileNetV2 模型是专门为移动和嵌入式设备设计的网络架构，该架构能在保持类似精度的条件下显著的减少模型参数和计算量，因为本实验最后需要把模型部署到手机上，所以选择了这个网络。mobilenet 的原始论文：https://arxiv.org/abs/1801.04381

MindSpore Lite 介绍

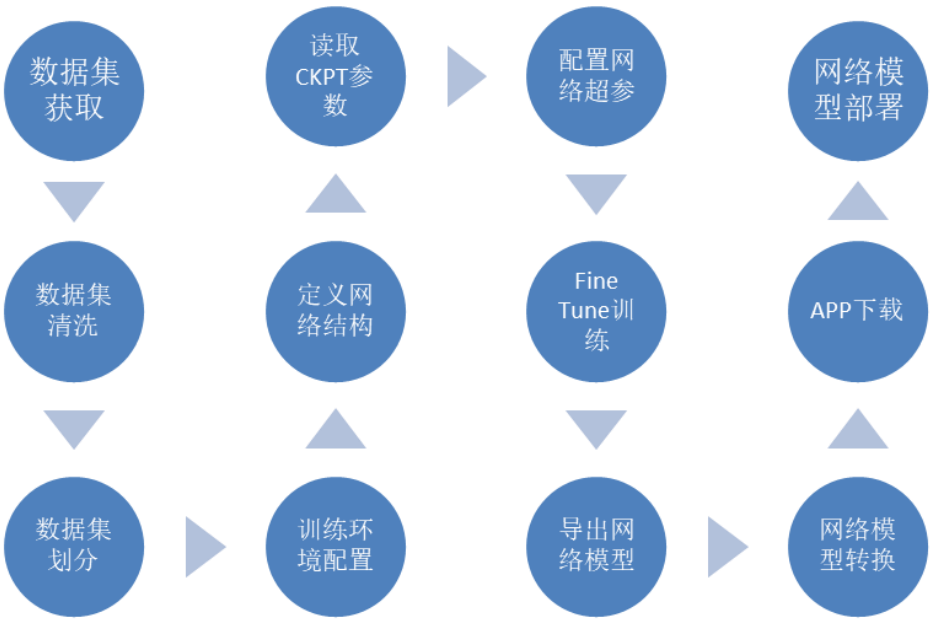
2020 年 3 月，MindSpore 全场景 AI 计算框架对外开源，作为端边云全场景的一部分，端侧领域 MindSpore Lite 在 2020 年 9 月正式发布并对外开源。

MindSpore Lite 支持 CPU/GPU/NPU，并且支持 IOS、Android 以及 LiteOS 等嵌入式操作系统，在模型方面支持 MindSpore/TensorFlow Lite/Caffe/Onnx 模型，极致性能、轻量化、全场景支持且高效部署。

更多详情，请参考 MindSpore Lite 官网：https://mindspore.cn/lite。

项目流程

本实验是一个完整的从数据获取、数据集清洗到端侧部署的全流程实验。整体实验流程如下图所示：



创新点与拓展工作

1. 我们将原来的猫狗识别，使用 Animal-10 数据集，将分类任务拓展到 10 种动物的分类。

2. 已更换多种其他图像类分类数据集、完成多种类别的多分类问题。

工作成果

猫狗分类

- 实验环境搭建

我们到华为云平台 ModelArt 上创建了实验用 Notebook，使用的配置为：

（硬件）Ascend: 1*Ascend910|CPU: 24 核 96GB

（软件）tensorflow1.15+mindspore1.7.0+cann5.1.0+euler2.8+aach64

在云平台上，环境已经搭建完毕，只需要输入以下指令激活环境

```
Bash
source /home/ma-user/anaconda3/bin/activate MindSpore
```

接下来，我们将实验文件上传至云平台上，得到的路径结构如下：

```
Bash
├─Workspace
│   ├─lost+found                  //自动生成文件夹
│   ├─code                        //Fine tune 训练代码
│   │   └─src                        //训练脚本调用的配置、数据处理等脚本
│   │   └─preprocessing_dataset.py   //预先处理数据集脚本
│   │   └─requirements.txt           //需要的依赖软件包
│   │   └─train.py                   //主训练脚本
│   │   └─mobilenetV2.ckpt           //预训练模型
│   │
│   ├─kagglecatsanddogs_3367a.zip //数据集
```

- 数据清洗

初始数据集往往存在很多的“脏数据”，这部分样本的存在会对模型产生不利的影响，如：影响准确率。所以，在实际工程中，往往第一步会进行数据清洗，本实验中，preprocessing_dataset.py 文件主要实现数据集清洗并划分数据集的功能。

```Bash
(MindSpore) [ma-user work]$cd code
(MindSpore) [ma-user code]$python
preprocessing_dataset.py ../kagglecatsanddogs_3367a.zip
extract dataset
extract dataset at /home/ma-user/work/code/dataset/PetImages
filter invaild images!
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/910.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/445.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/850.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/666.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/23.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/391.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/660.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/936.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Cat/140.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/663.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/50.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/296.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/719.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/565.jpg
```

```
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/561.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/414.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/522.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/543.jpg
Invalid image file, delete /home/ma-
user/work/code/dataset/PetImages/Dog/573.jpg
filter invaild images done, then split dataset to train and eval
final dataset at /home/ma-user/work/code/dataset/PetImages
```

由此，完成了对异常数据的清洗。

- 模型训练

首先，对代码进行修改使其能够正常运行：在 src/args.py line23 处作如下修改

```
Python
Before:
train_parser.add_argument('--dataset_path', type=str,
default="dataset\PetImages", help='Dataset path')
After:
train_parser.add_argument('--dataset_path', type=str,
default="dataset/PetImages", help='Dataset path')
```

输入以下命令进行训练：

```
Bash
python train.py
```

将生成的 mobilenetv2.mindir 转换成可移植的文件

```
Bash
call .\converter_lite.exe --fmk=MINDIR --
```

```
modelFile=.\mobilenetv2.mindir
 --outputFile=pet
```

由此生成 pet.ms 文件用于移植。

- 模型移植

将编写的对应的 APP 安装在手机上，并将生成的 pet.ms 文件移植到手机上。

多种动物分类

接下来我们基于 Animal-10 数据集进行多种动物的分类工作，仿照上述工作进行。

1. 数据集的获取，我们的数据集来源于 Kaggle:Kaggle Animal-10 Dataset，下载该数据集

2. 数据预处理：首先删去错误的文件，调用之前写好的脚本即可；第二步需要将所有图片转换成规定大小的图片(256*256)

a. 删去错误文件以及不符合要求的文件：调用已经写好的脚本

```Bash
python preprocessing_dataset.py ../pet10.zip
```

b. 图片大小转换与处理

```Python
import cv2

import os
from os import path

def image_pre_resize(config, dataset_path):
    """resize image size to image_height*image_width"""
    height = config["image_height"]
    width = config["image_width"]

    image_types = ["train", "eval"]
    for image_type in image_types:
        type_path = path.join(dataset_path,
f"{image_type}")
        animal_types = os.listdir(type_path)
        for animal in animal_types:
            animal_path = path.join(dataset_path,
f"{image_type}/{animal}")
            image_names = os.listdir(animal_path)
            for image in image_names:
                img =
cv2.imread(f"{dataset_path}/{image_type}/{animal}/{image}"
)
                resized_img = cv2.resize(img, (width,
height), interpolation= cv2.INTER_LINEAR)
                # write to image
```

```
    cv2.imwrite(f"{dataset_path}/{image_type}/{animal}/{image}
    ", resized_img)
```

3. 训练，我们使用 Ascend 硬件进行训练，输入指令如下，其余步骤同上

```Bash
python train.py --platform Ascend
```

4. 模型测试，我们可以编写以下代码对训练出来的模型进行测试，可以知道，该模型的正确率较高

```Python
import numpy as np
import mindspore
from mindspore import nn
from mindspore import Tensor
import cv2
import matplotlib.pyplot as plt

if __name__ == "__main__":
    # 读取 MINDIR 文件
    mindspore.set_context(mode=mindspore.GRAPH_MODE)
    graph = mindspore.load("code/mobilenetv2.mindir")
    model = nn.GraphCell(graph)

    # 图片读取与处理
    img_path = "code/dataset/PetImages/eval/gatto/1.jpeg"
    img_s = cv2.imread(img_path)
    h, w, c = img_s.shape
    new_w = int(500 / h * w)
    img_s = cv2.resize(img_s, (new_w, 500))
    img_s = cv2.cvtColor(img_s, cv2.COLOR_BGR2RGB)
    img_height = img_s.shape[0]
    img_width = img_s.shape[1]
    img = cv2.resize(img_s, (224, 224))
```

```
    mean = np.array([0.406 * 255, 0.456 * 255, 0.485 *
255]).reshape((1, 1, 3))
    std = np.array([0.225 * 255, 0.224 * 255, 0.229 *
255]).reshape((1, 1, 3))
    img = (img - mean) / std
    img = np.transpose(img, (2, 0, 1)).reshape((1, 3, 224,
224)).astype(np.float32)

    # 将图片输入模型计算概率
    output = model(Tensor(img)).asnumpy()
    softmax_output = np.exp(output)
    total = softmax_output.sum()
    softmax_output = softmax_output / total

    # 输出标签
    labels = ["cane", "cavallo", "elefante", "farfalla",
"gallina", "gatto", "mucca",       "pecora", "ragno",
"scolattolo"]
    labels[softmax_output.argmax()]
```

5. APP 使用端代码修改：由于原始 APP 只能显示“猫”或“狗”，我们这里在 Android Studio 中对 APP 代码进行修改，使其能够对十种动物显示进行支持（参考代码：MindSpore/ImageClassification）

由于模型输出未进行归一化，我们在代码中进行 Softmax 归一化再将结果作为置信度进行输出：

```
Java
String[] labels = new String[resultArray.length];
double[] scores = new double[resultArray.length];
int index = 0;
double total = 0;
for (String singleRecognitionResult:resultArray) {
    String[] singleResult = singleRecognitionResult.split(":");
    double score = Double.parseDouble(singleResult[1]);
    scores[index] = Math.exp(score);
```

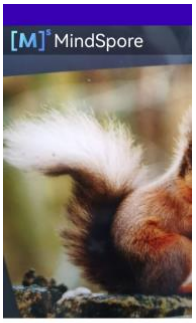
```
    total += scores[index];
    labels[index] = singleResult[0];
    index++;
}
// Softmax 处理
for(int i = 0;i < scores.length;i++){
    scores[i] = scores[i] / total;
}

for (int i = 0; i < resultArray.length;i++){
    double score = scores[i];
    String label = labels[i];
    if (score > 0.5) {
        recognitionObjectBeanList.add(new
RecognitionObjectBean(label, (float) score));
    }
}
Collections.sort(recognitionObjectBeanList, new
Comparator<RecognitionObjectBean>() {
    @Override
    public int compare(RecognitionObjectBean t1,
RecognitionObjectBean t2) {
        return Float.compare(t2.getScore(), t1.getScore());
    }
});
```

6. 最终在手机上得到的结果如下：

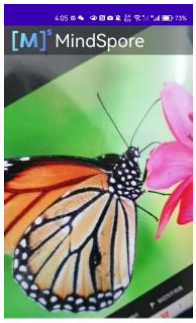

MindSpore
dog 63.65%
Inference Time: 38ms


MindSpore
horse 64.26%
Inference Time: 37ms


MindSpore
spider 56.46%
Inference Time: 51ms


MindSpore
squirrel 58.36%
Inference Time: 37ms


MindSpore
elephant 63.32%
Inference Time: 42ms


MindSpore
butterfly 90.19%
Inference Time: 47ms


MindSpore
cat 86.62%
Inference Time: 43ms


MindSpore
chicken 54.20%
Inference Time: 48ms


MindSpore
cow 57.67%
Inference Time: 42ms


MindSpore
sheep 54.39%
Inference Time: 47ms

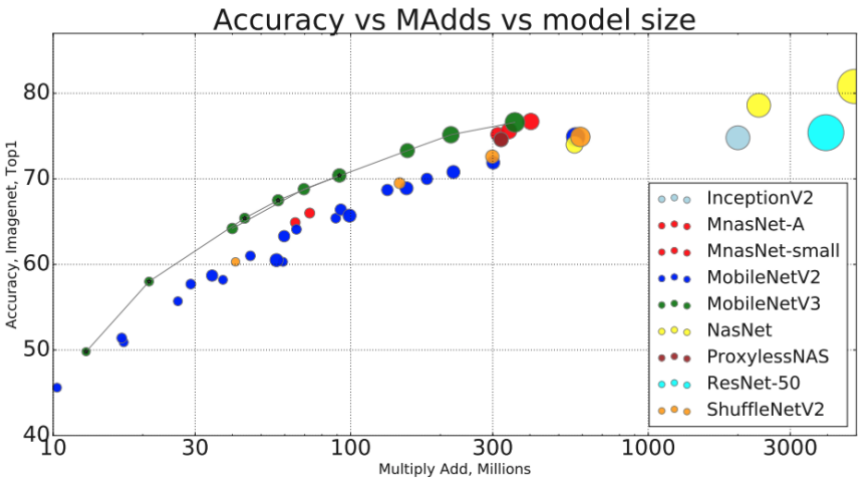
比较实验：分类模型

同样基于 Animal-10 数据集，我们尝试更换分类网络为 mobilenetV3，基于新的框架进行训练。

1. mobilenetV3 介绍

MobileNetV3 是由 google 团队在 2019 年提出的，其原始论文为 Searching for MobileNetV3。

我们知道 mobilenetV1 创造性的提出了深度可分离卷积，大大减少了模型的计算量；mobilenetV2 在第一版的基础上引入线性瓶颈和倒置残差结构。相比前两版，mobilenetV3 引入了注意力机制，采用了新的激活函数(h-swish)，重新设计耗时层结构。



在同等计算量下不同模型的规模和准确率的对比（摘自原始论文）

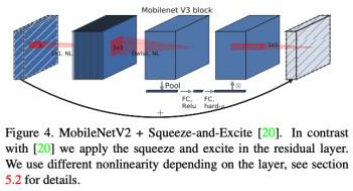


Figure 4. MobileNetV2 + Squeeze-and-Excite [20]. In contrast with [20] we apply the squeeze and excite in the residual layer. We use different nonlinearity depending on the layer, see section 5.2 for details.

注意力机制

Similarly, the hard version of swish becomes

$$\text{h-swish}[x] = x\frac{\text{ReLU6}(x+3)}{6}$$

hardversion-swish 激活函数

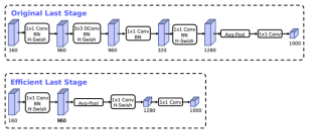


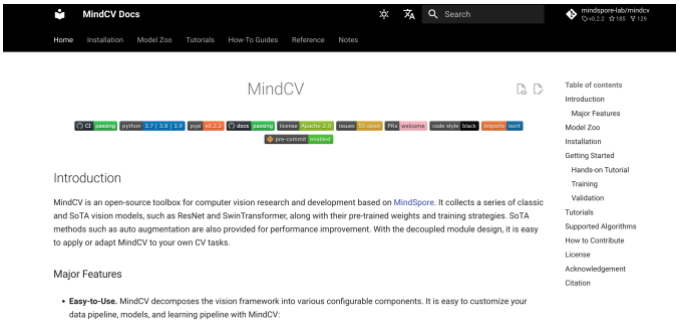
Figure 5. Comparison of original last stage and efficient last stage. This more efficient last stage is able to drop three expensive layers at the end of the network at no loss of accuracy.

重新设计的耗时层

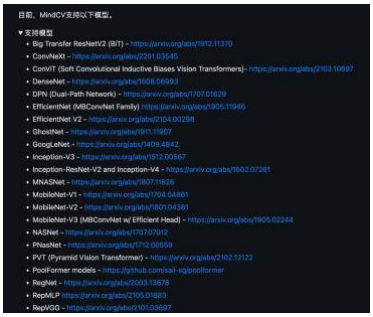
2. MindCV 库介绍

MindCV 是一个基于 MindSpore 开发的，致力于计算机视觉相关技术研发的开源工具箱。它提供大量的计算机视觉领域的经典模型和 SoTA 模型以及它们的预训练权重和训练策略。同时，还提供了自动增强等 SoTA 算法来提高模型性能。

MindCV 提供的模型脚本，为我们尝试各种不同的分类模型提供了便利。



MindCV 库简介



该库支持的模型

3. 训练过程

- 布置环境

与前述工作相似，我们新建华为云平台 ModelArt 上创建了实验用 Notebook，采用相同硬件参数，对于软件改用更新的 mindspore 版本（MindCV 需要 mindspore1.8+）

（硬件）Ascend: 1*Ascend910|CPU: 24 核 96GB

（软件）mindspore_1.10.0-cann_6.0.1-py_3.7-euler_2.8.3

进入虚拟平台，终端运行

```
Bash
pip install mindcv
```

- 布置仓库

将 MindCV 仓库部署到云平台

```Bash
git clone https://github.com/mindspore-lab/mindcv.git
```

更直接的方法是将仓库下载到本地后重新上传云平台。

- 数据处理

  数据处理部分同前述操作一致

在训练前的项目文件结构如下

```Bash
.
├── code
│   └── same_as_before
├── dataset
│   └── pet10
└── mindcv-main
    ├── configs
    ├── train.py
    └── validate.py
```

- 训练模型
  - 对`/mindcv-main/configs/mobilenetV3/mobilenet_v3_small_ascend.yaml`做如下修改

```YAML
# dataset
dataset: 'imagenet'
data_dir: 'dataset/pet10'
shuffle: True
dataset_download: False
batch_size: 75
drop_remainder: True
val_split: 'eval'

# model
model: 'mobilenet_v3_small_100'
```

```
num_classes: 10
pretrained: True
ckpt_path: ""
# ckpt_path: "ckpt/mobilenet_v3_small_100_best.ckpt"
keep_checkpoint_max: 10
ckpt_save_dir: './ckpt'
epoch_size: 30
dataset_sink_mode: True
amp_level: 'O3'
decay_epochs: 3
```

- 在终端中运行如下命令

```
Bash
python mindcv-main/train.py --config mindcv-
main/configs/mobilenetv3/mobilenet_v3_small_ascend.yaml --
device_target GPU --val_while_train --val_interval=1
```

脚本运行后会在根目录下生成 ckpt 文件夹，其中保存了训练过程中生成的模型 checkpoint 文件如最佳模型 mobilenet_v3_small_100_best.ckpt

- 测试模型准确率

运行 MindCV 脚本检验模型准确率

```
Bash
python mindcv-main/validate.py --config mindcv-
main/configs/mobilenetv3/mobilenet_v3_small_ascend.yaml
```

```
Model: mobilenet_v3_small_100
Num batches: 79
Start validating...
batch: 79/79, time: 51.916004s
{'Top_1_Accuracy': 0.887993862677407, 'Top_5_Accuracy': 0.9873417721518988, 'loss': 0.7901543665535843}
```

最终准确率展示

- 模型格式转换

在 mindcv-main 文件夹下新建 convert.py 文件如下

```Python
import mindspore as ms
import mindspore.nn as nn
from mindspore import Model
from mindcv.data import create_dataset, create_loader,
create_transforms
from mindcv.loss import create_loss
from mindcv.models import create_model
from mindcv.utils import ValCallback
from mindspore import export, Tensor
import numpy as np
from config import parse_args  # isort: skip

def load_model(args):
    ms.set_context(mode=args.mode)
    # read num clases
    num_classes = dataset_eval.num_classes() if
args.num_classes is None else args.num_classes
    # create model
    network = create_model(
        model_name=args.model,
        num_classes=num_classes,
        drop_rate=args.drop_rate,
        drop_path_rate=args.drop_path_rate,
        pretrained=args.pretrained,
        checkpoint_path=args.ckpt_path,
        ema=args.ema,
    )
    network.set_train(False)

    return network


if __name__ == '__main__':
    args = parse_args() # get args
```

```
    model = load_model(args) # load model mobilenetV3 we
trained
    input_tensor = Tensor(np.ones([1, 3, 224,
224]).astype(np.float32)) # define input size
    export(model, Tensor(input_tensor),
file_name='mobilenetv3', file_format='MINDIR') # export
model in mindir format
```

运行后再根目录下得到 mobilenetv3.mindir

4. 部署与结果展示

模型转换和部署到手机同上述步骤，最终结果类似。

比较实验：不同数据

我们采取了不同的数据集，使用同一模型进行训练，最终试验结果如下表所示：

数据集名称	描述	来源	Acc
动物分类	10 种动物分类	华为平台	99.1
口罩检测	戴/不戴口罩分类	MO 平台	99.9
明星脸识别	明星照片分类	MO 平台	99.0
beans	植物类型识别	Huggingface.co/beans	98.5

比较实验：不同模型

不同针对动物数据集训练相同批（30epoch）并测试，在华为平台上不同模型的分类准确率和响应时间分别是：

模型	准确率/%		响应时间/s	参数量/M
	top-1	top-5		
mobilenet v2(1.0)	85.5	99.1	50.8	3.54
mobilenet v3(small_1.0)	88.8	98.7	51.9	2.55

mobilenetv3 在分类 top1 准确率上有显著提升，在 top5 和响应时间方面几乎与 mobilenetv2 保持一致，在模型参数上远远小于 mobilenetv2。

三、总结与展望

本实验基于 MindSpore 深度学习框架，通过 Fine Tune 训练网络模型，再将模型转换为可以部署到手机端的 ms 模型，进而在手机端成功部署，实现模型在端侧的应用。通过此实验，用户对于 MindSpore 本地环境的搭建、MindSpore 框架的应用以及 MindSpore Lite 框架会有进一步的了解和掌握。

实验感想

通过这次实验，我对于人工智能与深度学习在具体任务上的应用有了进一步的认识与操作经验。我们在华为提供的平台上搭建实验模型，并更换数据集、更换模型进行对比实验与分类测试，并成功在本地与移动端两端完成了项目的部署，取得了出色的成效。本次项目经历将在日后更深层次的学习上起到更大作用。

——沈骏一

这次实验让我学习了如何使用华为平台进行模型训练以及移动端的部署。在实验中我尝试使用了新的深度学习工具 mindspore 和 mindcv，多番调试后成功复现，再不同的数据集和网络中运行流畅。这次实验锻炼了我的项目能力、拓展了深度学习网络的知识，也让我对我国人工智能产业发展信心倍增。

——朱语

这次实验我学习了利用华为 Mindspore 平台进行机器学习的训练，并通过猫狗分类的简单案例带我熟悉了一遍深度学习模型中从数据获取，到模型建立，到训练，部署的全过程及使用的工具，并尝试在一个较为复杂的问题中运用。这次实验让我对于人工智能与机器学习的技术掌握更加深入，运用能力更强。

——宋孟炫

项目分工

姓名	学号	主要工作	贡献占比
朱语	3200104139	模型拓展、实验测试、报告撰写、展示等	33.33%
沈骏一	3200100259	数据集拓展、实验测试、报告撰写等	33.33%
宋孟炫	3200105210	移动端搭建、实验测试、报告撰写等	33.33%

参考文献

1. Sandler M, Howard A, Zhu M, et al. Mobilenetv2: Inverted residuals and linear bottlenecks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 4510-4520.

2. Howard A, Sandler M, Chu G, et al. Searching for mobilenetv3[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 1314-1324.

3. Tan M, Le Q. Efficientnet: Rethinking model scaling for convolutional neural networks[C]//International conference on machine learning. PMLR, 2019: 6105-6114.

4. Zhang H, Wu C, Zhang Z, et al. Resnest: Split-attention networks[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022: 2736-2746.

5. Dai Z, Liu H, Le Q V, et al. Coatnet: Marrying convolution and attention for all data sizes[J]. Advances in neural information processing systems, 2021, 34: 3965-3977.

6. Guo M H, Lu C Z, Liu Z N, et al. Visual attention network[J]. arXiv preprint arXiv:2202.09741, 2022.

7. Si C, Yu W, Zhou P, et al. Inception transformer[J]. Advances in Neural Information Processing Systems, 2022, 35: 23495-23509.

8. Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, inception-resnet and the impact of residual connections on learning[C]//Proceedings of the AAAI conference on artificial intelligence. 2017, 31(1).