

Cat_Dog_Classification

1. 项目简介

基于MindsSpore框架实现了一个简单的二分类猫狗分类器，使用ResNet50网络进行训练，ONNX进行模型部署，并通过PyQt5进行简单的UI界面展示

2. 环境配置

本项目基于Linux环境进行训练得到权重文件并导出ONNX，在Win11进行模型部署和测试。将环境配置文件编写为Shell脚本，即：env.sh文件，通过如下命令进行环境部署

```
1 | bash env.sh
```

ONNX文件（放于项目根目录下）：

链接：<https://pan.baidu.com/s/17xlq1hValsjosvCEy0CgJw?pwd=2qpo>

提取码：2qpo

CheckPoint模型权重文件（放于项目根目录下）：

链接：<https://pan.baidu.com/s/1Q2Fhbb3dcSG9CQT0DVWsNw?pwd=0qnr>

提取码：0qnr

数据集（放于DataSet/PetImages目录下）：

Kaggle[猫狗数据集](#)

3. 目录介绍

```
1  ``
2  |-- cnn101.py
3  •   |-- cnn152.py
4  •   |-- cnn50.py
5  •   |-- env.sh
6  •   |-- export.py
7  •   |-- file_list.txt
8  •   |-- infer.py
9  •   |-- onnx_infer.py
```

```
10 • |-- process_data.py
11 • |-- README.md
12 • |-- resnet.py
13 • |-- resnet101.py
14 • |-- resnet152.py
15 • |-- train.py
16 • |-- train_256.py
17 • |-- train_adam.py
18 • |-- train_attention.py
19 • |-- train_cnn.sh
20 • |-- train_continue.py
21 • |-- train_test.py
22 • |-- train_transfer.py
23 • |-- val.py
24 • |-- dataset
25 • |   |-- PetImages
26 • |       |-- clean.py
27 • |       |-- partition.py
28 • |       |-- Cat
29 • |       |-- Dog
30 • |-- log
31 • |   |-- file_list.txt
32 • |   |-- file_path.py
33 • |   |-- cnn101_lr0.1_bs64
34 • |       |-- train.log
35 • |   |-- cnn152_lr0.1_bs64
36 • |       |-- train.log
37 • |   |-- cnn50_lr0.1_bs64
38 • |       |-- train.log
39 • |   |-- resnet101_lr0.1_bs64
40 • |       |-- train.log
41 • |   |-- resnet152_lr0.1_bs64
42 • |       |-- train.log
43 • |   |-- resnet50_attention_lr0.1_bs256
44 • |       |-- train.log
45 • |   |-- resnet50_lr0.001_bs64
46 • |       |-- train.log
47 • |   |-- resnet50_lr0.001_opt-adam_bs64
48 • |       |-- train.log
49 • |   |-- resnet50_lr0.01_bs256
50 • |       |-- train.log
51 • |   |-- resnet50_lr0.01_bs64
52 • |       |-- train.log
53 • |   |-- resnet50_lr0.01_opt-adam_bs64
54 • |       |-- train.log
55 • |   |-- resnet50_lr0.1_bs256
56 • |       |-- train.log
57 • |   |-- resnet50_lr0.1_bs64
58 • |       |-- train.log
59 • |-- model_utils
60 • |   |-- config.py
61 • |   |-- config
```

```
62 • | | |-- resnet101_imagenet2012_config.yaml
63 • | | |-- resnet152_imagenet2012_config.yaml
64 • | | |-- resnet18_cifar10_config.yaml
65 • | | |-- resnet18_cifar10_config_gpu.yaml
66 • | | |-- resnet18_imagenet2012_config.yaml
67 • | | |-- resnet18_imagenet2012_config_gpu.yaml
68 • | | |-- resnet34_imagenet2012_config.yaml
69 • | | |-- resnet50_cifar10_config.yaml
70 • | | |-- resnet50_imagenet2012_Ascend_Thor_config.yaml
71 • | | |-- resnet50_imagenet2012_Boost_config.yaml
72 • | | |-- resnet50_imagenet2012_config.yaml
73 • | | |-- resnet50_imagenet2012_GPU_Thor_config.yaml
74 • | | |-- resnet_benchmark_GPU.yaml
75 • | | |-- se-resnet50_imagenet2012_config.yaml
76 • | |-- __pycache__
77 • | |-- config.cpython-37.pyc
78 • | |-- config.cpython-39.pyc
79 • |-- output
80 • |-- plot_log
81 • | |-- file_list.txt
82 • | |-- plot_log.py
83 • | |-- csv_data
84 • | | |-- cnn101_lr0.1_bs64
85 • | | | |-- training_data.csv
86 • | | | |-- cnn152_lr0.1_bs64
87 • | | | |-- training_data.csv
88 • | | | |-- cnn50_lr0.1_bs64
89 • | | | |-- training_data.csv
90 • | | |-- resnet101_lr0.1_bs64
91 • | | | |-- training_data.csv
92 • | | |-- resnet152_lr0.1_bs64
93 • | | | |-- training_data.csv
94 • | | |-- resnet50_attention_lr0.1_bs256
95 • | | | |-- training_data.csv
96 • | | |-- resnet50_lr0.001_bs64
97 • | | | |-- training_data.csv
98 • | | |-- resnet50_lr0.001_opt-adam_bs64
99 • | | | |-- training_data.csv
100 • | | |-- resnet50_lr0.01_bs256
101 • | | | |-- training_data.csv
102 • | | |-- resnet50_lr0.01_bs64
103 • | | | |-- training_data.csv
104 • | | |-- resnet50_lr0.01_opt-adam_bs64
105 • | | | |-- training_data.csv
106 • | | |-- resnet50_lr0.1_bs256
107 • | | | |-- training_data.csv
108 • | | |-- resnet50_lr0.1_bs64
109 • | | | |-- training_data.csv
110 • | |-- img
111 • | |-- cnn101_lr0.1_bs64
112 • | | |-- training_validation_plot.png
113 • | |-- cnn152_lr0.1_bs64
```

```

114 • | | |-- training_validation_plot.png
115 • | | |-- cnn50_lr0.1_bs64
116 • | | |-- training_validation_plot.png
117 • | | |-- resnet101_lr0.1_bs64
118 • | | |-- training_validation_plot.png
119 • | | |-- resnet152_lr0.1_bs64
120 • | | |-- training_validation_plot.png
121 • | | |-- resnet50_attention_lr0.1_bs256
122 • | | |-- training_validation_plot.png
123 • | | |-- resnet50_lr0.001_bs64
124 • | | |-- training_validation_plot.png
125 • | | |-- resnet50_lr0.001_opt-adam_bs64
126 • | | |-- training_validation_plot.png
127 • | | |-- resnet50_lr0.01_bs256
128 • | | |-- training_validation_plot.png
129 • | | |-- resnet50_lr0.01_bs64
130 • | | |-- training_validation_plot.png
131 • | | |-- resnet50_lr0.01_opt-adam_bs64
132 • | | |-- training_validation_plot.png
133 • | | |-- resnet50_lr0.1_bs256
134 • | | |-- training_validation_plot.png
135 • | | |-- resnet50_lr0.1_bs64
136 • | | |-- training_validation_plot.png
137 • |-- __pycache__
138 • | |-- resnet.cpython-39.pyc
139 ``

```

4. 代码说明

DataSet:

`clean.py` 为数据清洗脚本，保证图片格式为JPEG格式，删除其他非法格式

`partition.py` 为数据集划分脚本，9:1划分训练集和验证集

`pachong.py` 为爬虫程序，在Win11环境下的个人PC中，通过添加Chrome驱动，修改查询词，即可将对应的图片下载到本地

Train:

`train.py` 为训练脚本，定义超参，其他脚本文件类似，不过是修改了部分参数

`export.py` 为导出脚本，将训练好的best.ckpt文件导出ONNX用于下一步的模型部署和UI展示

Log:

该部分为模型训练过程的日志保存文件夹

Output:

该部分为模型权重文件保存文件夹

Plot_Log:

该部分为模型训练的Loss和Acc数据处理和可视化部分，通过运行plot_log.py创建img文件夹和csv文件夹，保存可视化结果和对应的数据

Inner:

infer.py 可进行推理，但没有UI界面

部署+UI:

onnx_infer.py 该部分最好在Win个人PC中运行（本人在服务器中没有root权限，环境没法配置），需要下载onnxruntime、PyQt5

```
1 | pip install onnxruntime
2 | pip install PyQt5
```

UI界面展示:



Demo:

见Demo文件夹

5. 项目总结

本项目比较基础的实现了一个有UI界面的二分类ResNet网络分类器，主要在训练网络过程中尝试了改进网络结构、调整超参、进行图像增强、加入Attention机制等方式，同时熟悉了ONNX部署模型，后续可作为Web后端做成一个完整的项目，并通过uni-app打包实现多平台部署。

水平有限，能力不足，欢迎issue、fork和star，给个免费的小星星！