

# 2024 《人工智能导论》大作业

(可参考修订)

任务名称: 暴力图像二分类检测模型及其接口的实现与实验

完成组号: 7

小组人员: 邓迪、苏煥泽

完成时间: 2024-06-17

## 1. 任务目标

基于暴力图像检测数据集，构建一个检测模型。该模型可以对数据集的图像进行不良内容检测与识别。

模型是二分类模型，具有一定的泛化能力：不仅能够识别与训练集分布类似的图像，对于 AIGC 风格变化、图像噪声、对抗样本等具有一定的鲁棒性。

为便于他人使用模型，我们实现了一个接口类。其提供的函数接受 `tensor` 向量化的 `n` 张固定大小图片，并输出以 `n` 长数组表示的分类结果。

## 2. 具体内容

### 2.1. 实施方案

#### 2.1.1. 模型实现

我们利用文档里提供的几部分代码，成功进行了模型的代码实现，并进行了一些调参优化工作，以实现更好性能。

第一次实验的参数如下：

`max_epochs=10`

`batch_size = 128`

`lr = 3e-4`

模型测试结果：

```
Testing DataLoader 0: 100%|██████████| 12/12 [00:12<00:00, 1.06s/it]
```

Test metric	DataLoader 0
test_acc	0.9860140085220337

第二次实验的参数如下：

`max_epochs=40`

`batch_size = 128`

`lr = 3e-4`

模型测试结果：

```
Testing DataLoader 0: 100%|██████████| 12/12 [00:12<00:00, 1.07s/it]
```

Test metric	DataLoader 0
test_acc	0.9650349617004395

进程已结束,退出代码0

第三次实验的参数如下：

`max_epochs=20`

`batch_size = 128`

`lr = 3e-4`

模型测试结果：



接口的构造函数首先接受 GPUid、已训练模型文件的位置(ckpt\_root+ckpt\_model), batch 大小, 以此定义接口使用的硬件、模型和接收图片大小。

接口的分类函数 classify 则接受 imgs (一个大小为  $n*3*224*224$  的 tensor), 随后 model.eval() 设置模型为评估模式, 遍历 imgs 中的所有图片, 计算模型的输出量。通过 torch.max 取得 output 中的分类结果, 类型转换后存储到 preds 中。

### 2.2.2. main.py 使用实例分析

以下是接口类的使用实例之一。

```
1 import torch
2
3 import classify
4 from dataset import CustomDataModule
5
6 if __name__ == '__main__':
7     batch_size = 128
8     data_module = CustomDataModule(batch_size=batch_size)
9     data_module.setup()
10    imgs = torch.ones(batch_size, 3, 224, 224)
11    for i in range(0, batch_size, 1):
12        imgs[i] = data_module.test_dataset.__getitem__(i + 128)[0]
13    model = classify.ViolenceClass(gpu_id=[0], ckpt_root="C:/Users/HUAWEI/Desktop/AI/train_logs/",
14                                   ckpt_model="resnet18_pretrain_test/version_0/checkpoints/resnet18_pretrain_test"
15                                   "-epoch=20-val_loss=0.04.ckpt",
16                                   batch_size=batch_size)
17    ans = model.classify(imgs)
18    print(ans)
```

首先, 实例指定了输入图片的数量 batch\_size, 并调用 dataset.py 中的函数建立了数据集 data\_module, 将其初始化。随后, 使用数据集的既有方法得到 imgs (一个大小为  $n*3*224*224$  的 tensor), 作为标准格式的输入。

随后, 调用接口类的构造函数, 指定 GPUid、已训练模型文件位置, batch 大小。

调用接口类的分类函数, 得到标准格式的结果并输出。

## 3. 工作总结

### 3.1. 收获、心得

#### 3.1.1. 邓迪心得

本次实验中, 我进行了接口类的实现工作, 这对于我是首次考虑模型的易用性问题。在过去浅尝辄止的尝试中, 我满足于模型的训练和测试, 而没有考虑其如何被使用, 也没有考虑其具体的结果产生过程。

这次大作业一方面让我进行了模型接口设计、基于 github 平台的代码共享等崭新的实践, 另一方面也促使我不满足于对模型模块的拼凑, 而更加深入地理解了模型的运作。

#### 3.1.2. 苏煥泽心得

第一次成功训练模型并有好的测试结果的这次尝试, 相当于打开人工智能领域的大门, 看到了这个领域怎么做以及能得到什么结果, 虽然只是一次小的尝试, 但对于模型的训练与测试有了初步的认识。希望以后能继续深入的学习与实践。

### 3.2. 遇到问题及解决思路

#### 3.2.1. 人员短缺问题

由于项目开始时间较晚, 叠加期末复习临近, 我们小组出现了人头数和单人工时的双重短缺。项目因而出现了人力上的困难。

对此，我们一方面倒排工期，制定严格时间表和分工；另一方面充分发挥个人能动性，见缝插针进行项目代码的设计开发。

### 3.2.2. 训练速度问题

由于组员邓迪的计算机未安装独立显卡，只能使用 `cpu` 作为训练的设备，因而出现训练缓慢的问题。一次训练往往需要耗费七至八个小时，严重影响了迭代开发、问题发现与解决的效率。

在组员苏煥泽完成了 `cuda` 驱动的配置，开始使用其电脑的 `NVIDIA` 显卡进行训练后，情况有了明显改观。一次训练（10 轮）仅需十分钟即可完成，大大加速了我们完善代码的工作。

由此我们深刻认识到，显卡硬件对于人工智能模型学习的重要性。

### 3.2.3. 虚拟内存问题

```
File "D:\4399\Anaconda\envs\test4\lib\site-packages\pytorch_lightning\callbacks\checkpoint_callback.py", line 10, in <module>
    from pytorch_lightning.callbacks.base import Callback
File "D:\4399\Anaconda\envs\test4\lib\site-packages\pytorch_lightning\callbacks\checkpoint_callback.py", line 11, in <module>
    import torch
File "D:\4399\Anaconda\envs\test4\lib\site-packages\torch\tensor.py", line 1, in <module>
    raise err
SError: [WinError 1455] 页面文件太小，无法完成操作。 Error loading "D:\4399\Anaconda\envs\test4\lib\site-packages\torch\tensor.py"
```

运行环境所在的文件夹在计算机 D 盘中，查找发现计算机默认情况下没有给 D 盘分配虚拟内存。程序运行没有分配虚拟内存，就遇到了上面的问题。

对此，我们在高级系统设置中给 D 盘分配虚拟内存，问题也就得以解决。

## 4. 课程建议

希望能在课堂具体讲或者介绍大作业相关的实践操作，在理论教学的课堂中加上人工智能实践相关的作业或练习。

大作业亲自尝试训练大模型很有趣也很有学习的价值，希望以后的课程能接触更多更新颖的模型。