

# 2024人工智能导论大作业

任务名称: 不良内容图像检测

完成组号:3

小组人员: 翁解语 孔珺晓 毛锐

姓名	学号	分工
翁解语	522031910417	实验环境配置 模型分析与调试 数据集爬取与处理 报告中实施方案&代码分析部分撰写 github 项目管理 工作分配与汇总
孔珺晓	522031910341	爬虫代码 数据集爬取与汇总 服务器显卡提供 loss可视化绘图
毛锐	522020910051	报告中工作总结, 课程建议部分撰写 数据集爬取

完成时间: 2024年6月21日

## 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, 并达到较高准确率. 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有合理的运行时间。

## 2. 具体内容

### (1) 实施方案

按照调试代码的时间顺序来讲述我们的方案:

- 根据老师给出的代码新建程序, 配置好conda环境, 确认gpu可用,微调代码以适应windows环境, 不调整超参数直接开始第一轮训练. 将训练结果应用于训练集上测试, 可以发现正确率非常高.

```
F:\anaconda3\envs\violence_check\lib\site-packages\pytorch_lightning\trainer\connectors\data_connector.py:436: Consider setting 'persistent_workers=True' in 'test_dataloader' to speed up the dataloader
worker initialization.
Testing DataLoader 0: 98% | 60/61 [00:13<00:00, 4.44it/s]F
:anaconda3\envs\violence_check\lib\site-packages\torch\nn\modules\conv.py:456: UserWarning: Plan failed with a cudnnException: CUDNN_BACKEND_EXECUTION_PLAN_DESCRIPTOR: cudnnFinalize Descriptor Failed
cudnn_status: CUDNN_STATUS_NOT_SUPPORTED (Triggered internally at C:\actions-runner\work\pytorch\pytorch\builder\windows\pytorch\aten\src\aten\native\cudnn\Conv_v8.cpp:919.)
return F.conv2d(input, weight, bias, self.stride,
Testing DataLoader 0: 100% | 61/61 [00:13<00:00, 4.49it/s]

Test metric      DataLoader 0
-----
test_acc          0.9993548393249512

(violence_check) PS F:\violence_check>
```

在训练集上达到这个准确率往往是过拟合了的结果. 于是在网上查找新的真实图片作为测试集(共约100张), 结果如下图:

```
Testing DataLoader 0: 0% | 0/1 [00:00<?, ?it/s]F
:anaconda3\envs\violence_check\lib\site-packages\torch\nn\modules\conv.py:456: UserWarning: Plan failed with a cudnnException: CUDNN_BACKEND_EXECUTION_PLAN_DESCRIPTOR: cudnnFinalize Descriptor Failed
cudnn_status: CUDNN_STATUS_NOT_SUPPORTED (Triggered internally at C:\actions-runner\work\pytorch\pytorch\builder\windows\pytorch\aten\src\aten\native\cudnn\Conv_v8.cpp:919.)
return F.conv2d(input, weight, bias, self.stride,
Testing DataLoader 0: 100% | 1/1 [00:00<00:00, 3.88it/s]

Test metric      DataLoader 0
-----
test_acc          0.7368420958518982

(violence_check) PS F:\violence_check>
```

由上图,我们的模型可能有些过拟合, 导致训练集上正确率很高而其他图片上正确率一般.

### 2. 解决模型的过拟合问题

为了解决过拟合问题, 我们分析代码并尝试使用以下方法来提高模型在新测试集上的性能.

- 改用更深, 性能更好的模型resnet50(虽然更大的模型会导致过拟合情况加剧):

```
self.model = models.resnet50(pretrained=True)
```

- 正则化: 使用L2正则化项, 在优化器中设置 `weight_decay`, 调整参数后得到以下结果:

```
def __init__(self, num_classes=2, learning_rate=1e-3, weight_decay=2e-6):
    #具体超参需要调整
    super().__init__()
    self.model = models.resnet18(pretrained=True)
    num_ftrs = self.model.fc.in_features
    self.model.fc = nn.Sequential(
        nn.Dropout(0.5),
        nn.Linear(num_ftrs, num_classes)
    )

    self.learning_rate = learning_rate
    self.weight_decay = weight_decay
    self.loss_fn = nn.CrossEntropyLoss()
    self.accuracy = Accuracy(task="multiclass", num_classes=2)
```

dropout: 虽然原模型最后一层全连接层前添加了一个 Dropout, 但可以尝试在其他层也使用这一方法:

- ```
x = self.model.layer1(x)
x = F.dropout(x, p=0.5, training=self.training)
x = self.model.layer2(x)
x = F.dropout(x, p=0.5, training=self.training)
x = self.model.layer3(x)
x = F.dropout(x, p=0.5, training=self.training)
x = self.model.layer4(x)
x = F.dropout(x, p=0.5, training=self.training)
```

- 早停法: 让训练在5个回合内不优化的情况下停下来:

```
# 设置早停法回调
early_stopping_callback = EarlyStopping(
    monitor='val_loss',
    patience=5, # 如果验证损失在5个epoch内没有改善, 则停止训练
    mode='min',
    verbose=True
)
```

经过上述优化, 我们的模型可以在真实图片测试集上达到以下效果:

```
e) when logging on epoch level in distributed setting to accumulate the metric across devices.
Testing DataLoader 0: 100% |██| 1/1 [00:00<00:00, 2.69it/s]

Test metric      DataLoader 0
-----
test_acc         0.8666666746139526
```

并且在AI生成的测试集上也可以达到不错的效果:

```
Testing DataLoader 0: 100% |██| 1/1 [00:00<00:00, 3.05it/s]

Test metric      DataLoader 0
-----
test_acc         0.7857142686843872
```

### 3. 加强模型对于对抗样本的鲁棒性

- 首先我们需要生成对抗样本来测试模型目前对于对抗的性能. 从原测试集中选取约200张图片, 直接进行测试; 使用对抗算法对这些图片进行处理, 原图片和新图片的准确率分别如下图所示:

| Testing DataLoader 0: 100% <div></div>   2/2 [00:00<00:00, 5.28it/s] |                    |
|----------------------------------------------------------------------|--------------------|
| Test metric                                                          | DataLoader 0       |
| test_acc                                                             | 0.995121955871582  |
|                                                                      |                    |
| Test metric                                                          | DataLoader 0       |
| test_acc                                                             | 0.4390243887901306 |

可以看到我们的对抗算法十分有效, 得到了一个对于二分类问题较差且较为混沌的结果.

- 为了加强模型对于对抗样本的鲁棒性, 我们将对抗样本加入到训练集中并重新训练, 并在不同源的对抗样本上测试结果, 如下图所示:

| Testing DataLoader 0: 100% <div></div>   1/1 [00:00< |                    |
|------------------------------------------------------|--------------------|
| Test metric                                          | DataLoader 0       |
| test_acc                                             | 0.7560975551605225 |

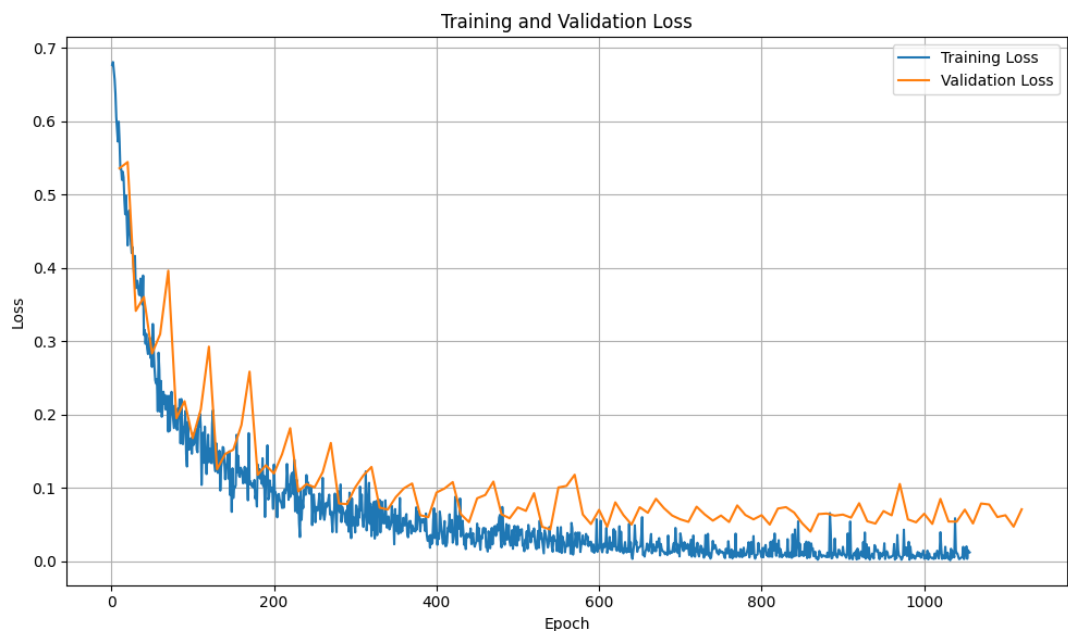
效果较为显著.

#### 4. 获得更多优质训练集

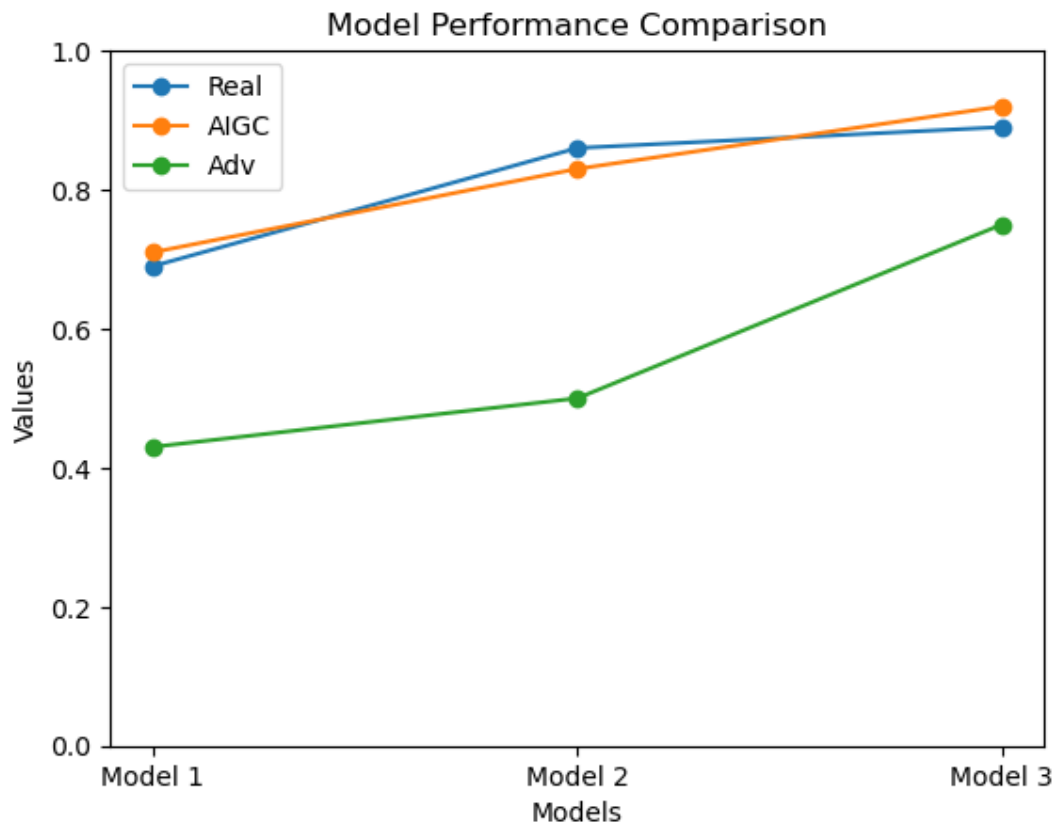
通过以上优化步骤, 我们的模型优化遇到了瓶颈. 此时我们选择从网上大量爬取AIGC与带噪声的数据集并将其加入训练集来提升模型性能. 加入训练集的图片合集链接详见本报告附录.

#### 5. 分析模型优化过程

1. trainloss和valloss随步数下降情况如下图所示:



2. 老师给的原始称为模型1, 优化过后但是不修改训练集的模型称为模型2, 训练集加入对抗样本内容后的模型称为模型3. 这三个模型在不同于训练集的测试集上准确率分别如下图所示:



## (2) 核心代码分析

在train方面, 我们修改的代码均已在上一个部分提供了代码分析. 除此之外, 一些接口方面的操作也比较重要:

- 将训练和测试图片统一resize到3\*224\*224. 因为大多数训练图片都是这个尺寸, 归一化这一点保证了模型的准确性较少的受到图片尺寸的影响:

```
• if split == "train":
    self.transforms = transforms.Compose([
        transforms.Resize((224, 224)),
        .....
    ])
else:
    self.transforms = transforms.Compose([
        transforms.Resize((224, 224)),
        transforms.ToTensor(),
    ])
```

- 添加脚本入口点. 这一点保证了在模块被直接运行时执行特定的代码, 而在模块被作为库导入时不会执行这些代码. 在调试过程中可以更好的适应各种环境:

```
• if __name__ == '__main__':
    main()
```

- 记录train\_loss和val\_loss, 并将其可视化, 方便调试模型:

```
• self.train_loss_file = open("train_loss.txt", "a")
  self.val_loss_file = open("val_loss.txt", "a")
```

## 3. 工作总结

- (1) 收获、心得

- **加深对各种分类模型的理解：**在项目中，我们深入探索了ResNet50和ResNet18等图像分类模型，通过应用并微调这些预训练模型，提升了识别准确率并深入理解模型对特定数据集的适应性。我们比较了不同模型的性能和泛化能力，提升了模型选择和优化的技能。
- **认识AI伦理相关问题：**在处理不良内容检测中，我们探讨了AI伦理问题，如数据隐私和算法偏见。我们意识到在数据采集和使用上需遵守法律，保护用户隐私，关注模型的偏见，确保公正性和透明性。
- **提升解决问题的能力, 感受团队协作的重要性;**

## (2) 遇到问题及解决思路

### 1. 过拟合问题

- **描述:** 模型在训练集上表现优异，但在新的测试集上准确率较低，表明存在过拟合问题。
- **解决思路:**
  - **模型迁移:** 使用更复杂的网络结构如ResNet50，提升模型复杂度和学习能力。
  - **正则化应用:** 引入L2正则化减少模型复杂度，帮助减轻过拟合。
  - **增加Dropout:** 在网络多层中添加Dropout，随机关闭部分神经元，提升泛化性。
  - **早停法:** 如果验证集损失在多个epoch后未改善，则提前终止训练。

### 2. 对抗样本鲁棒性不足

- **描述:** 模型对于对抗样本分类效果差，显示出鲁棒性不足。
- **解决思路:**
  - **对抗样本生成:** 利用多种对抗攻击算法生成大量对抗样本，评估模型的当前鲁棒性。
  - **对抗训练:** 将生成的对抗样本纳入训练集，通过对抗训练方法提高模型对这些样本的识别能力。

### 3. 数据集不足问题


- **描述:** AIGC拒绝生成符合要求的暴力内容图像，导致暴力图像类别样本不足。
- **解决思路:**
  - **网页爬虫技术:** 应用爬虫从网络上抓取符合条件的暴力图像，以补充数据集。
  - **人工筛选:** 对爬取的图像进行严格筛选，确保质量与相关性。
  - **数据增强:** 对补充的暴力图像进行数据增强处理，增加样本多样性，提升模型泛化能力。

## 4. 课程建议

- **增加实践教学:** 可以考虑在课程中加入更多的小型实践项目，这样可以帮助我们更好地理解相关理论在实际中的应用。
- **提供预习指南:** 在相关实践作业开始前，提供一些与实践相关的预习指南，帮助我们更好地理解作业要求，减少完成过程中的困难。
- **早点布置大作业:** 大作业可以在学期初或期中布置，让我们可以早点开始做。

附录: 我们所添加的图片数据网址:

2000张AIGC图片, 2000张adv图片

|                                                                                   |                                                                                                                                                                              |  |
|-----------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
|  | 分享内容: <a href="https://jbox.sjtu.edu.cn/l/w1gOPw">violence check新增训练集&amp;ckpt文件</a> 链接地址: <a href="https://jbox.sjtu.edu.cn/l/w1gOPw">https://jbox.sjtu.edu.cn/l/w1gOPw</a> |  |
|                                                                                   |                                                                                                                                                                              |  |
| 来自于: 翁解语                                                                          |                                                                                                                                                                              |  |