Lab2. CNN

- TA 姚文浩 -

whyao23@m.fudan.edu.cn

大纲

- 简答题和计算题（30分）
- 代码填空题（70分）
  - 任务简介
  - 作业要求
  - 提交方式

大纲

- **简答题和计算题（30分）**
- 代码填空题（70分）
  - 任务简介
  - 作业要求
  - 提交方式

简答题和计算题（共30分）

第一题（10 分）

对于一张输入大小为224*224，通道数为3的图像，我们想使用大小为3*3的卷积核进行运算，输出一个64通道的特征图。

1) 每个卷积核的通道数为多少？**（2 分）**

2) 共需要多少个卷积核？ **（2 分）**

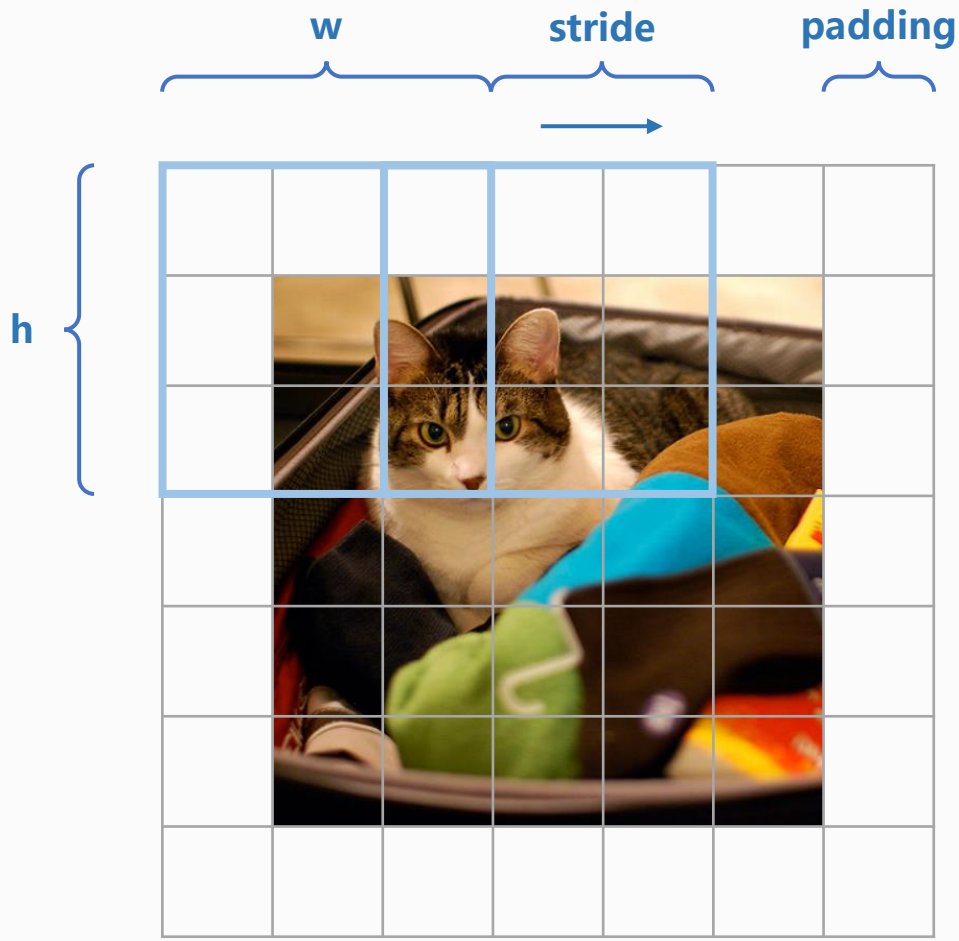
3) 若设置图像的padding size为1，卷积核步距（stride）为2，那么输出的特征图的长和宽为多少？ **（3 分）**

4) 该层网络的参数量为多少？（不考虑 bias） **（3 分）**

简答题和计算题（共30分）

回顾：卷积核输出特征图的形状计算

- 卷积核相关参数
  - 宽、高　w, h
  - 步距　stride
  - 填充　padding
- 输出图像大小计算
  - $W' = \left\lfloor \frac{W+2p-w}{s} \right\rfloor + 1$
  - $H' = \left\lfloor \frac{H+2p-h}{s} \right\rfloor + 1$



w=3, h=3, s=2, p=1

W=5, H=5

简答题和计算题（共30分）

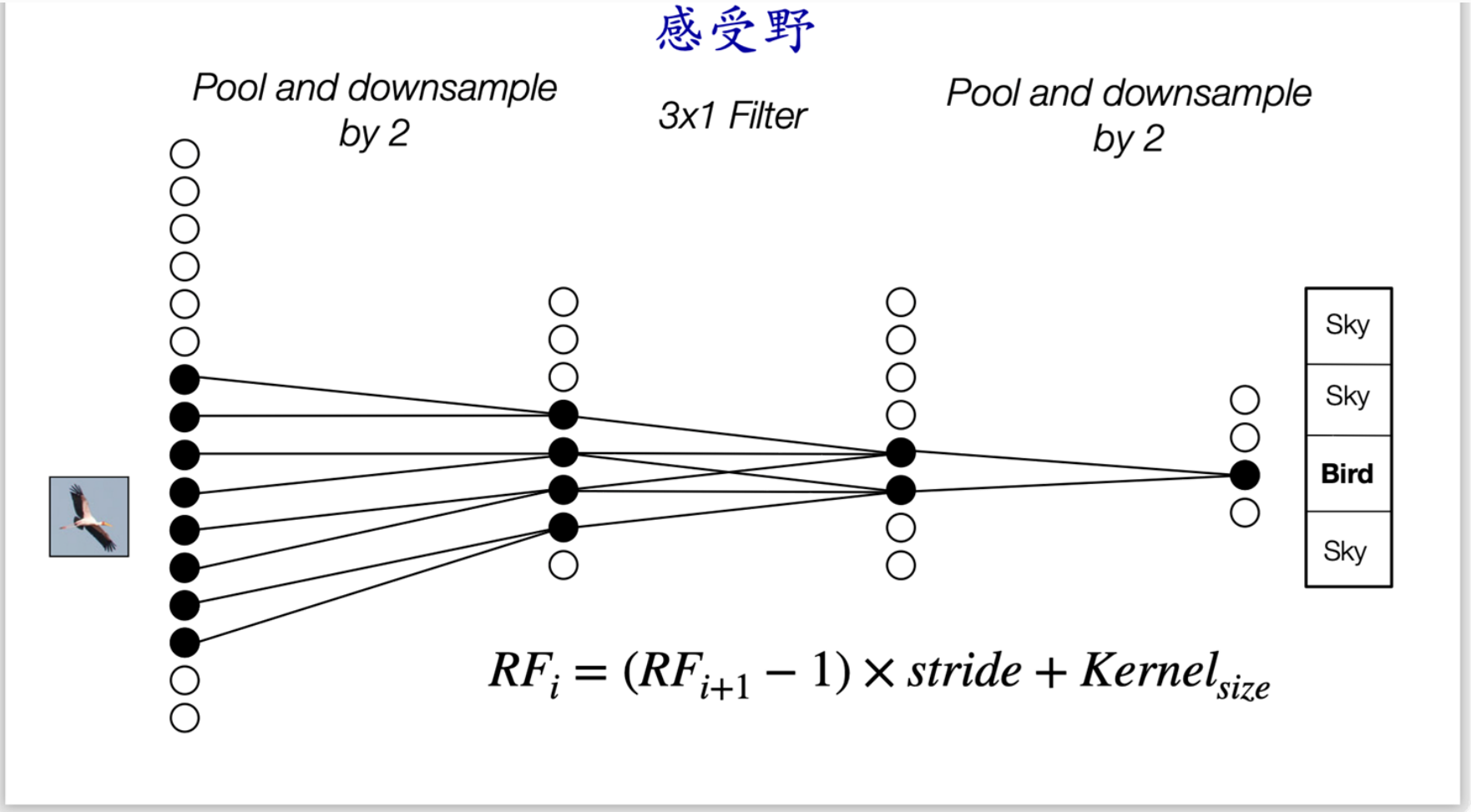
第二题（10 分）

下侧表格描述了一个卷积神经网络的网络结构。求输出层的感受野（形状为二维）。

Layer	Kernel Size	Stride
Input		
Conv1	3*3	1
Pool1	2*2	2
Conv2	3*3	1
Pool2	2*2	2
Conv3	3*3	2
Conv4	3*3	1
Conv5	3*3	1
Output		

简答题和计算题（共30分）

回顾：感受野计算



简答题和计算题（共30分）

第三题（10 分）

在VGG16、ResNet等网络中，卷积核的长宽通常都是奇数而非偶数，这样的设计可能是出于什么方面的考虑？ （思考题，言之有理即可，尽量展开讨论）

大纲

- 简答题和计算题（30分）
- **代码填空题（70分）**
  - **任务简介**
  - 作业要求
  - 提交方式

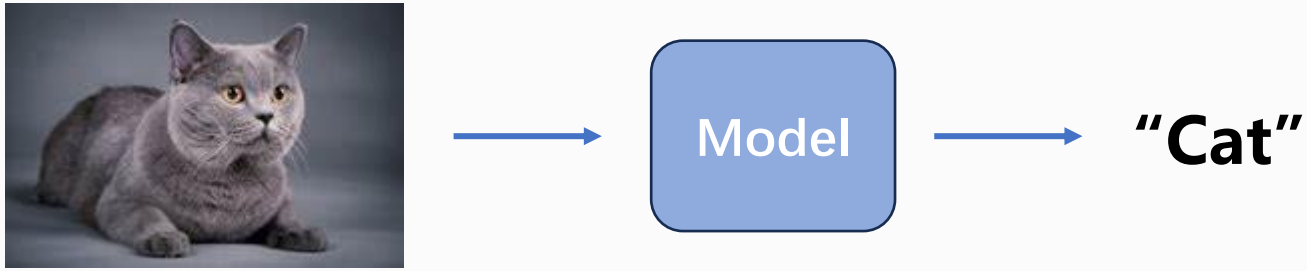
作业2——任务简介

- **作业链接**
- **Kaggle 的使用方式可参考 Lab1 的 ppt**

- Kaggle 每周有固定的 gpu 使用时间，同学们每次使用完之后要记得关闭环境，以免造成时间浪费
- Kaggle 后台运行方法：
  https://www.yuque.com/wjpoom/fudan-ai/qe1zxrx6l4sgltyg?singleDoc#
  《Kaggle使用手册02：后台运行》
  密码：arg3

作业2——任务简介

- 使用卷积神经网络 ResNet-34 实现图像分类任务



作业2——任务简介

- 数据集：**Animals-10**
  - 共有 10 个动物类别
  - 约 2.8 万张图片，随机选取了一个子集（¼）作为本次作业的数据
  - 对于子集再次进行划分，训练集:验证集:测试集=7:1:2
    - 训练集包含 4236 张图片
    - 验证集包含 605 张图片
    - 测试集包含 1211 张图片

作业2——任务简介

- 评价指标：分类正确率 Accuracy

$$\mathrm{Acc} = \frac{sum(\mathrm{pred} == \mathrm{label})}{len(\mathrm{data})} \times 100\%$$

大纲

- 简答题和计算题（30分）
- **代码填空题（70分）**
  - 任务简介
  - **作业要求**
  - 提交方式

任务 1 ——实现数据增强

- 数据增强：样本数量不足或者样本质量不够好时，通过一定的方式改变输入数据，以生成更多样的训练样本，从而提高模型的泛化能力和效果，避免过拟合
- 例如在图像分类任务中，对于输入的图像可以进行一些简单的**平移、缩放、颜色变换**等操作，这些操作不会改变图像的类别，但可以增加训练样本的数量

任务 1 ——实现数据增强

- PyTorch 对数据增强的实现

```
from torchvision.transforms import v2

transform = v2.RandomCrop(size=(224, 224))
out = transform(img)

plot([img, out])
```

- 同时应用多种数据增强
  - transforms = v2.Compose([...])
- 参考资料
  - pytorch transform tutorial
  - 各种数据增强的 api 文档

任务 1 ——实现数据增强

- 代码补全（**数据增强后输出图片大小必须为 3*224*224**） **(10 分)**

```
train_tfm = v2.Compose([
    v2.Resize((224,224)),

    # ========== 请在此处自行补充一些数据增强方式 ==========

    # ====================================================

    v2.ToTensor(),
    v2.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])
```

- 效果验证（输出 5 张不一样的图片即可）

√

×

任务 2 ——补全数据集 __getitem__ 函数

- Pytorch 数据集和数据加载都通过类来实现
  - torch.utils.data.Dataset: 存储了数据及其对应标签
  - torch.utils.data.DataLoader: 将数据集存储的数据以 batch 的形式迭代输出

```
from torch.utils.data import Dataset

class MyDataset(Dataset):
    def __init__(self, file):
        self.data = ...

    def __getitem__(self, index):
        return self.data[index]

    def __len__(self):
        return len(self.data)
```

Read data & preprocess

Returns one sample at a time

Returns the size of the dataset

任务 2 ——补全数据集 __getitem__ 函数

- 代码补全（10 分）

```
def __getitem__(self, idx):

    """
    以下是一段空缺的代码，你需要对 AnimalDataset 类的 __getitem__ 函数进行补充。
    目标:
      根据参数 idx 从数据集的 path_lst 和 label_lst 中取出对应的图片路径 fname 和标签 label
    """
    # ========== 请在此处补充你的代码 ==========

    # ========================================

    img = Image.open(fname)
    img = self.transform(img)

    return img, label
```

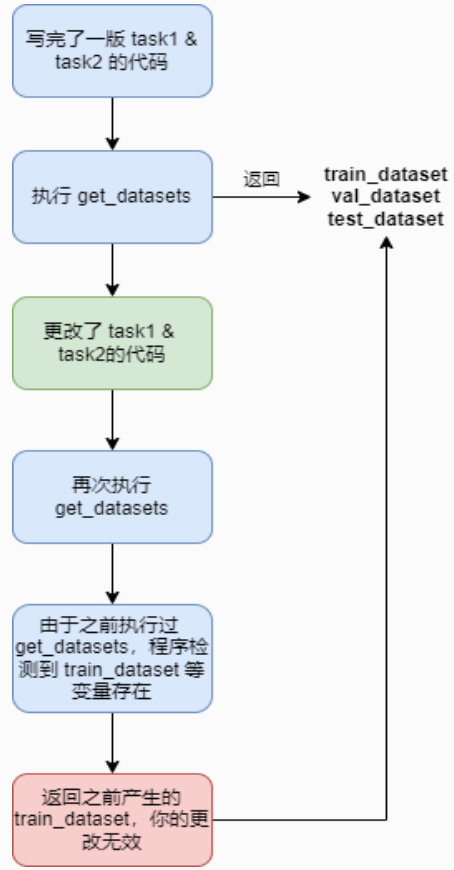
任务 2 ——补全数据集 __getitem__ 函数

- 验证

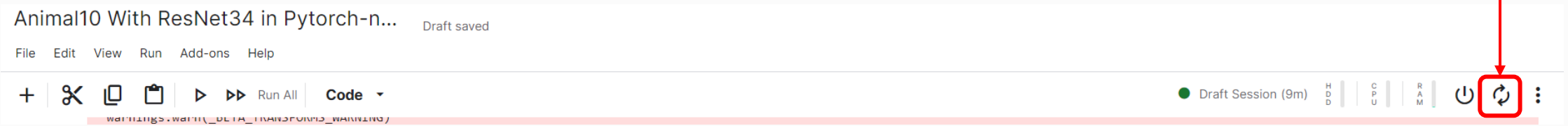
```
correct_labels = train_dataset.label_lst[:50]
your_labels = [train_dataset[_][1] for _ in range(50)]

if your_labels == correct_labels:
    print("__getitem__ 函数验证正确! ")
else:
    print("__getitem__ 函数验证错误! ")

print("")

img = train_dataset[0][0]
if img.size() == torch.Size([3, 224, 224]):
    print("数据增强输出形状正确! ")
else:
    print("数据增强输出形状错误! ")
```

注意事项

```python
def get_datasets(data_dir, sample_ratio=0.25, split_ratio=[0.7, 0.1, 0.2]):
    """
    按照一定比例（7:1:2），手动将数据集划分为训练集、验证集和测试集。
    由于之前设定了随机数种子，因此每位同学得到的训练集、验证集和测试集都是一样的。这个函数无需进行修改。
    """
    assert sum(split_ratio) == 1.0
    global train_dataset, val_dataset, test_dataset
    try:  # 保证函数多次执行时结果不变
        if train_dataset is not None:
            print("datasets exist!")
            return train_dataset, val_dataset, test_dataset
    except NameError:
        pass
```

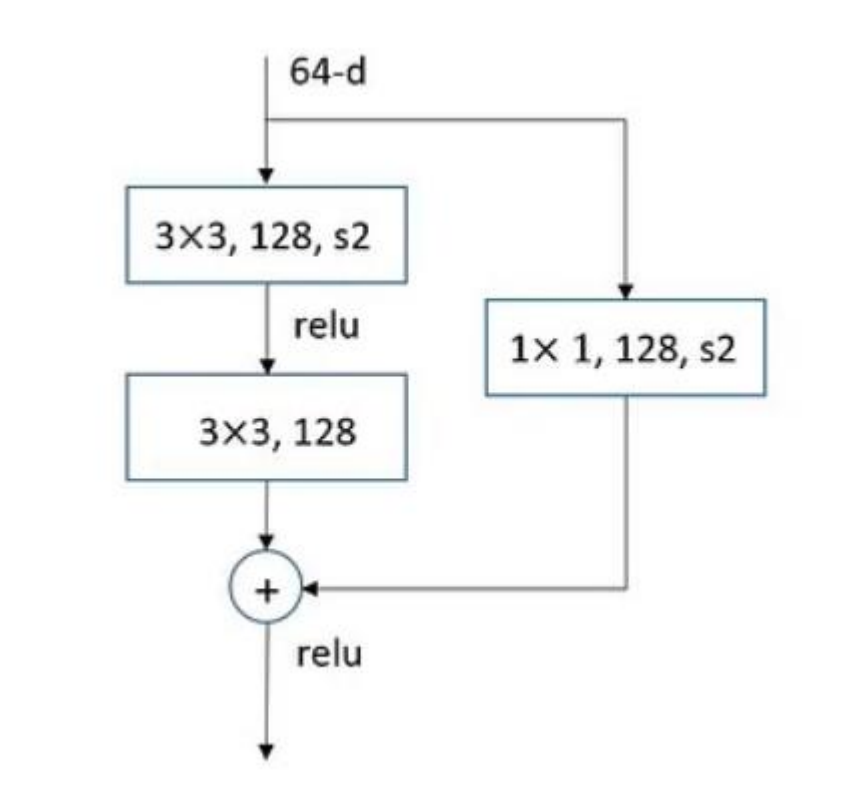


- 如果没有蓝框代码，多次执行 get_datasets 仍然会导致产生不一样的划分结果
- 蓝框代码：防止多次执行该函数导致数据集图片不一样。但副作用是：如果中途更改了任务 1 和 任务 2 的代码，可能无法产生更新后的 datasets
- 怎么办?
- 如果你的任务 1 和任务 2 的代码需要修改，那么你更改之后，需要**重启环境**



任务 3 ——补全 ResidualBlock 类的 forward 函数

- 残差块 ResidualBlock
  - 卷积运算
  - 残差连接



任务 3 ——补全 ResidualBlock 类的 forward 函数

- 代码补全
  (10 分)

```
def forward(self, x):
    """
    残差块的前向传播函数。输入 x 会通过 2 层卷积，得到的输出会与原输入进行残差求和。
    """

    """
    以下是一段空缺的代码，你需要对残差块类的 forward 函数进行补充。
    目标:
      1. 将输入 x 用变量 residual 存储起来，防止输入丢失
      2. 将输入 x 分别通过 2 层卷积层，得到输出 out
      3. 将经过卷积层的输出 out 与输入 residual 相加得到新的 out
      注意: 如果 self.downsample 不为空，则需要在相加前将输入先用 self.downsample 函数处理。别忘了使用
    """
    # ========== 请在此处补充你的代码 ==========


    # ========================================

    return out
```

任务 3 ——补全 ResidualBlock 类的 forward 函数

- 输出形状验证

```
print("【以下测试用例用于检验tensor的形状，不保证值正确】")
for i in range(len(strides)):
    print(f"以下是第{i+1}个测试用例，输入形状为{tensor_sizes[i]}，期望输出形状为{tensor_sizes[i+1]}")
    x = torch.randn(tensor_sizes[i])
    x_out = rbs[i](x)
    if x_out.shape == torch.Size(tensor_sizes[i+1]):
        print(f"第{i+1}个测试用例通过！")
    else:
        tup = tuple(torch.tensor(x_out.shape).detach().cpu().tolist())
        print(f"第{i+1}个测试用例未通过！你的输出tensor形状为{tup}")

del in_channels, out_channels, strides, downsamples, rbs, tensor_sizes
```

任务 4 ——补全训练代码

- 神经网络训练步骤
  - 前向传播
  - 损失计算
  - 反向传播
    - zero_grad, backward, step
- 可参考作业 1 的 logistic regression 部分的训练代码
- 代码补充（**10 分**）

```
import gc

for epoch in range(num_epochs):
    with tqdm(train_loader, desc='Train [{}/{}]'.format(epoch+1
        for images, labels in t:
            # 将数据移动至 GPU
            images = images.to(device)
            labels = labels.to(device)


            """
            以下是一段空缺的代码，你需要对模型训练代码进行补充。
            训练过程包括:
            1. 前向传播
            2. 损失计算
            3. 反向传播
            """
            # ========== 请在此处补充你的代码 ==========

            # ========================================

            t.set_postfix(loss=loss.item())
            del images, labels, outputs
            torch.cuda.empty_cache()
```

任务 5 ——评价指标调优

- 根据**测试集**上的准确率指标给分
- 不能使用预训练模型，网络必须随机初始化
- 分段给分规则（计分时准确率四舍五入保留到小数点后2位，例如0.623计为0.62）

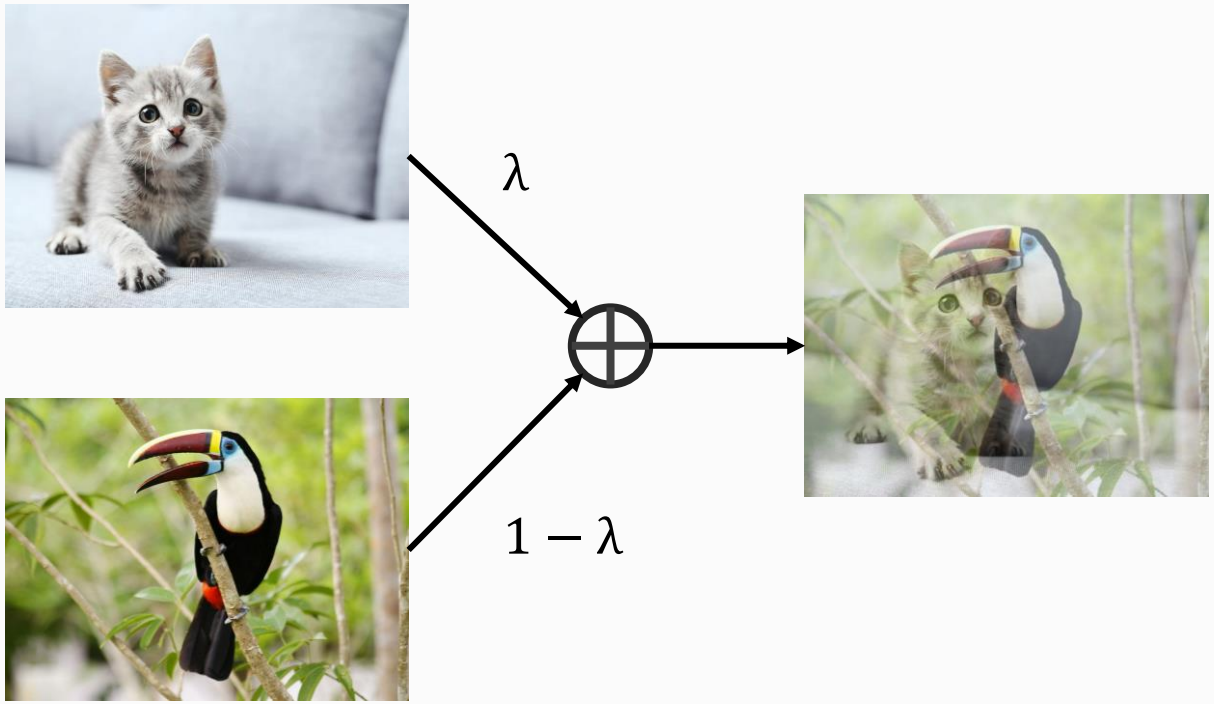
Baseline	Accuracy	Score
Vallina	[0.45, 0.63]	**15**
Simple	(0.63, 0.70]	**18**
Medium	(0.70, 0.77]	**20**
Hard	(0.77, 0.83]	**25**
Strong	(0.83, 1.00]	**30**

任务 5 提示

- 该任务难点
  - 数据量偏少
- 要增加准确率可以从什么角度入手?
  - 数据
  - 模型
  - 损失函数和优化器
  - 训练策略

任务 5 提示——数据

- 数据增强——选用多样化且合适的数据增强方式
  - horizontal flip, crop, rotate, color/light/contrast…
  - mixup(*)[1]



$$x = \lambda x_a + (1-\lambda)x_b$$

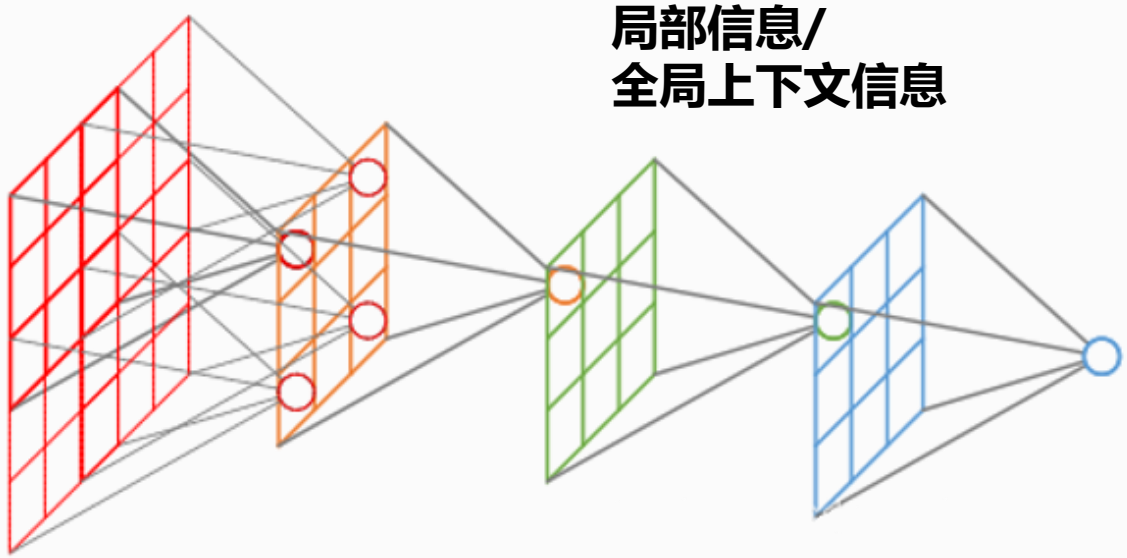
$$pred = \text{model}(x)$$

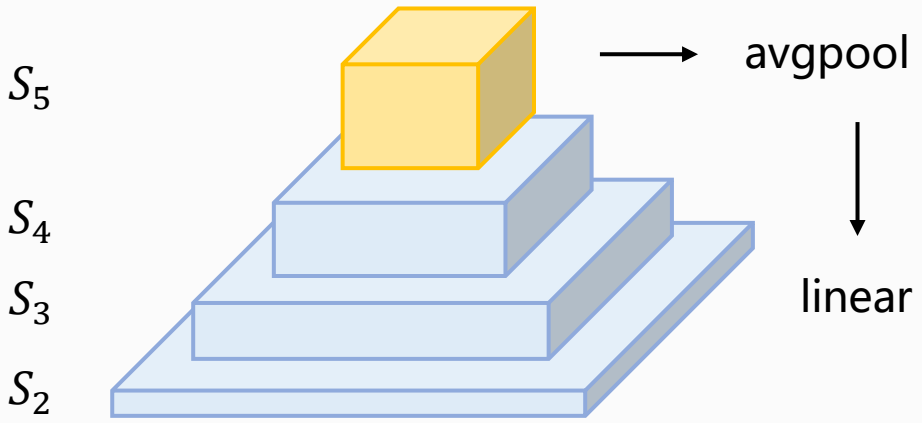
$$loss = \lambda \cdot \text{criterion}(pred, y_a) + (1-\lambda) \cdot \text{criterion}(pred, y_a)$$

[1] Zhang, Hongyi, et al. "mixup: Beyond empirical risk minimization." *arXiv preprint arXiv:1710.09412* (2017).

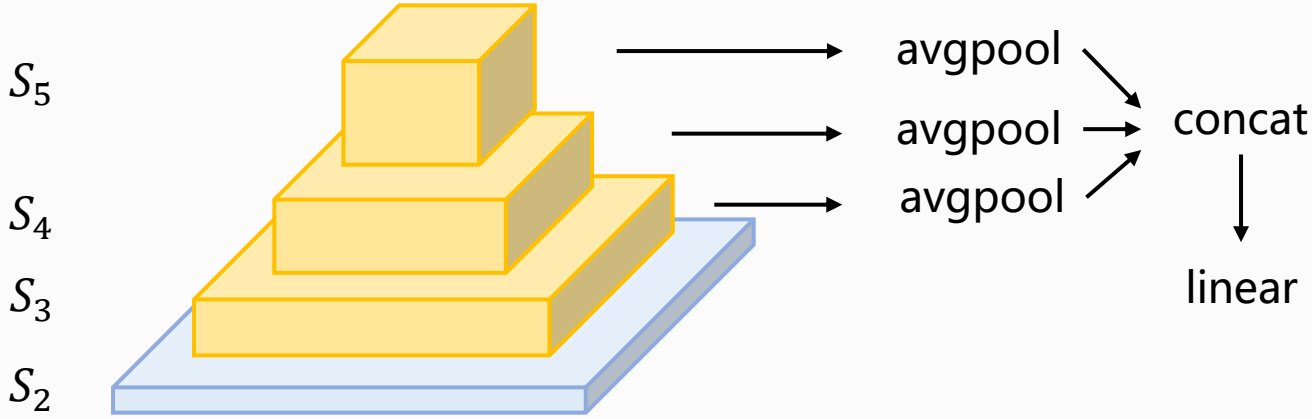
任务 5 提示——模型

- 使用其他预定义/更深层的模型
  - 参考资料：torchvision models, resnet
  - **weights=None / pretrained=False**
  - 模型更深意味着更久的训练时间
- 多尺度特征图（multi-scale features）





single-scale



multi-scale

任务 5 提示——模型

- 多尺度特征图（multi-scale features）
  - 可以修改 ResNet 类的 forward 函数
  - 建议另外写一个函数，这样可以不破坏原来的forward逻辑

```python
def forward(self, x):
    x = self.conv1(x)
    x = self.maxpool(x)
    x = self.conv2_x(x)
    x = self.conv3_x(x)
    x = self.conv4_x(x)
    x = self.conv5_x(x)

    x = self.avgpool(x)
    x = x.view(x.size(0), -1)
    x = self.fc(x)

    return x
```

```python
def forward(self, x):
    if self.multi_scale:
        return self.forward_multi_scale(x)

    x = self.conv1(x)
    x = self.maxpool(x)
    x = self.conv2_x(x)
    x = self.conv3_x(x)
    x = self.conv4_x(x)
    x = self.conv5_x(x)

    x = self.avgpool(x)
    x = x.view(x.size(0), -1)
    x = self.fc(x)

    return x
```

任务 5 提示——损失函数及优化器

- 损失函数
  - CrossEntropy
  - label smoothing（防止过拟合的策略）
- 优化器
  - SGD -> **Adam**, AdamW ...

任务 5 提示——训练策略

- 超参数调节
  - learning rate, epoch, batch size…
- 训练策略
  - **lr drop** (快收敛时将 lr 下降为0.1倍)
  - early stopping
  - …

- 多做实验，耐心

任务 5 提示——其他

- 同学们若有精力可以继续尝试以下方法：
  - Feature Pyramid Network (FPN)
  - **Test Time Augmentation**
  - Ensemble Model
  - …

大纲

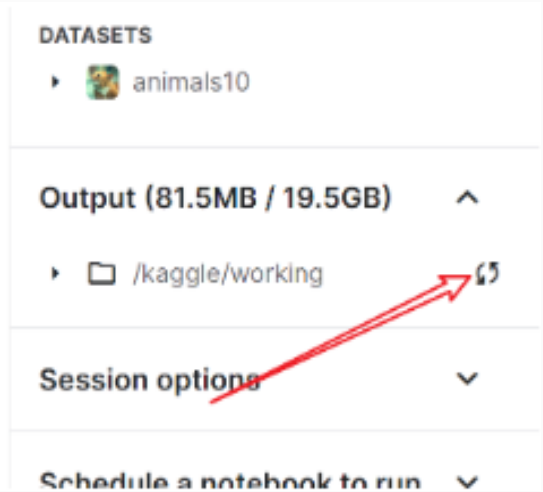
- 简答题和计算题（30分）
- 代码填空题（70分）
  - 任务简介
  - 作业要求
  - **提交方式**

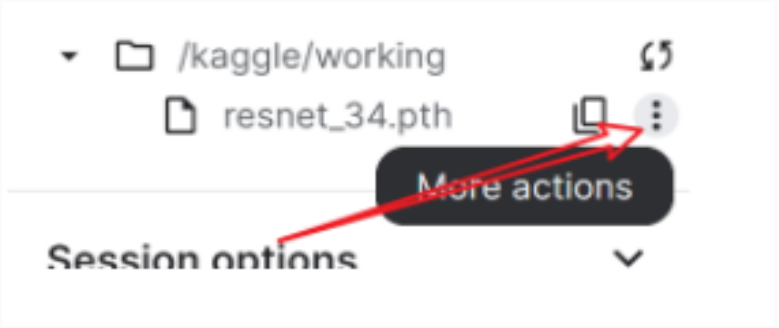
作业2——提交方式

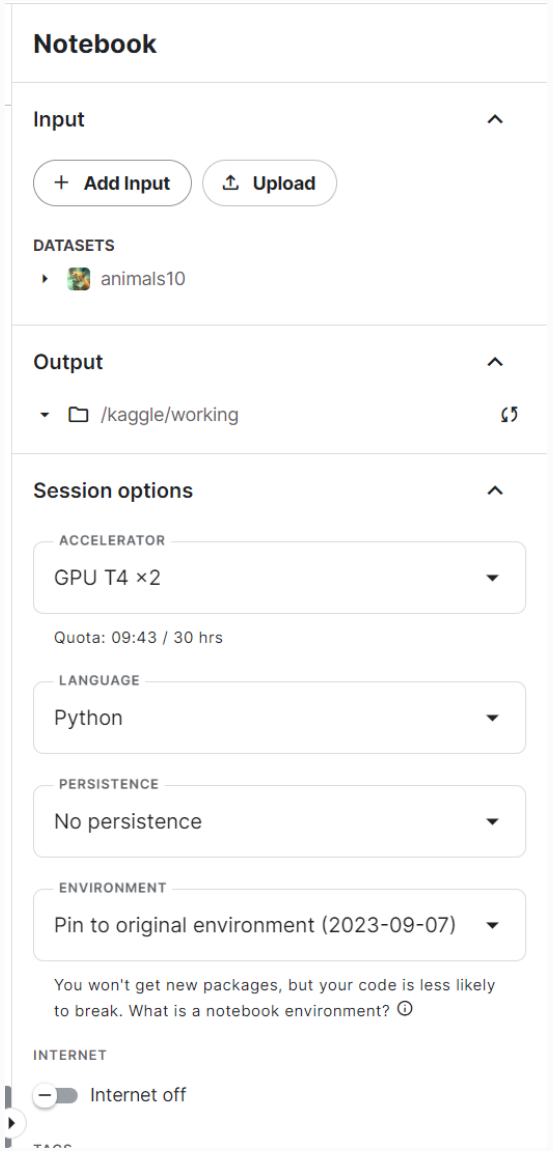
- 计算题和简答题
  - 用笔在稿纸上写出**答题过程**
  - 拍照，将其命名为：***学号_姓名_lab2_answer.jpg***
  - 在 eLearning 上提交
- 代码填空题
  - 在 Kaggle 上运行完实验后将 ipynb 文件下载 (File -> Download Notebook)
  - 重命名为：***学号_姓名_lab2.ipynb***
  - 在 eLearning 上提交

作业2——提交方式

- 代码填空题
  - 下载模型的操作如右图和下图
  - 模型重命名为：***学号_姓名_model.pth***
  - 作业截止后会抽取部分同学名单，被抽到的同学需要将模型文件发送至 whyao23@m.fudan.edu.cn







作业2——提交方式

- ddl：**2024.04.30 23:59**
- elearning 晚交一天倒扣 10 分
- 计算分数时以最后提交的版本为准（包括晚交的扣分）
- 若在完成作业的过程中遇到任何问题，均可通过各种方式提问
- **严禁抄袭，若发现抄袭者和被抄袭者本次作业均 0 分！**