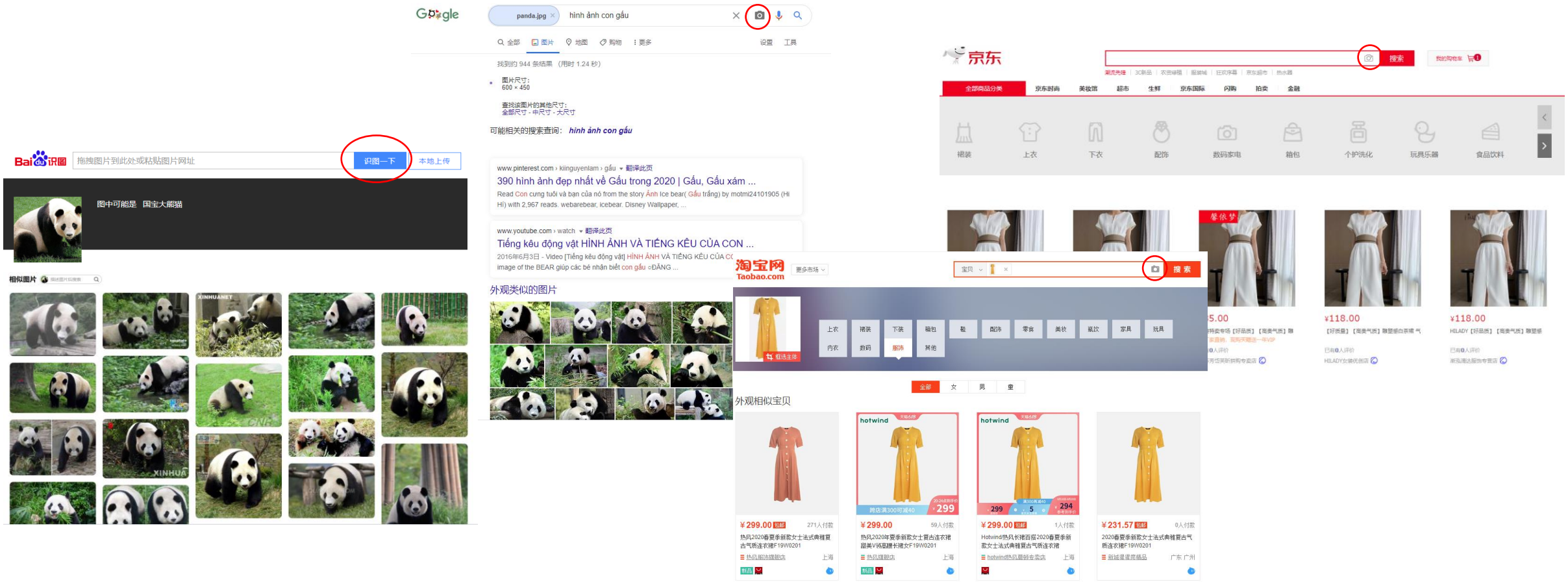
9. CNN

- 图像检索
- 卷积神经网络
- 使用 PyTorch 提取图像特征

图像检索

- 早期的图像检索技术研究主要是基于文本的图像检索（TBIR），通过对图片的文本描述来查找图像库中具有相应标签的图像。随着技术的发展，基于内容的图像检索（CBIR），即以图搜图也成为了各大网站的标配功能。



图像检索

- 以图搜图是如何实现的呢？以图搜图的核心其实就是在被检索的图像库（Library）中找到和参与检索的图片（Query）最相似的图片集合。那么问题就变成了如何定义图片之间的相似程度。

Query

a

b

c

d

e

f

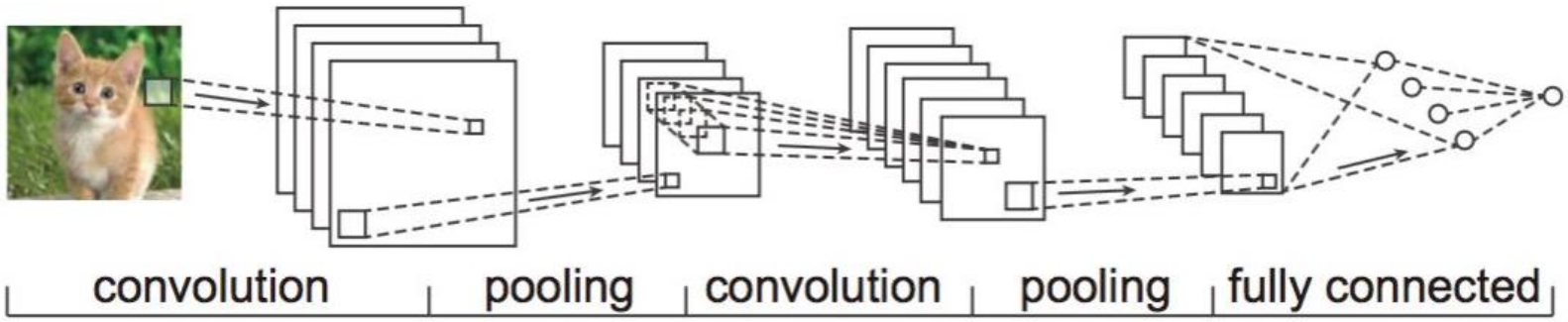
Library

图像检索

- 对于上页中的例子，与待检索图像（Query）最相近的应该是图像 f；但是图像库中的其他图片与 Query 的相似程度应该怎么排序呢？
- 如果从考虑图像颜色分布的情况来看，图像 b 和 e 也会有较高的相似程度；如果考虑交通工具这个范畴，图像 c 则会有较高的排序。
- 所以如果我们能从图像的内容中提取出相应的语义信息，如颜色，纹理，种类等等，那么这个以图搜图的检索任务就自然得到了解决。
- 深度学习模型通过一种端到端学习的范式，使得模型能够学习到图像的这种特征。所以我们可以使用模型来将图像转换成一个能够代表图像的向量，从而用向量之间的相似程度代表图像之间的相似程度，从而解决以图搜图的任务。

图像特征的提取

- 传统的图像特征提取有很多的方法，如 SIFT，HOG 等，我们不在本次实验课学习。
- 通过上一节实验课的学习，大家应该了解了一些深度学习的入门知识和 PyTorch 的使用方法。本节课我们将使用深度模型提取图像的特征，从而来帮助我们构建以图搜图的搜索引擎。



卷积神经网络

- 上节实验课我们简单介绍了什么是神经网络，并且使用了一种卷积神经网络 ResNet20 在 Cifar10 数据集上训练了一个图像分类的模型。现在我们来介绍一下什么是卷积神经网络。
- 卷积神经网络是一类包含卷积计算且具有深度结构的前馈神经网络 (Feedforward Neural Networks)，是深度学习 (deep learning) 的代表算法之一。其在图像处理的领域有着非常广泛的应用。
- 更详细的介绍请阅读下面的链接（希望大家都能认真学习一下，便于对后续实验内容的理解）：https://cs231n.github.io/convolutional-networks/

使用 PyTorch 提取图像特征

```
import torch
print(torch.hub.list('pytorch/vision'))
```

导入 torch 模块，看看 PyTorch官方给我们提供了哪些现成的深度模型。

```
Using cache found in /root/.cache/torch/hub/pytorch_vision_master
['alexnet', 'deeplabv3_resnet101', 'deeplabv3_resnet50', 'densenet121', 'densenet161', 'densenet169', 'densenet201', 'fcn_resnet101', 'fcn_resnet50', 'googlenet', 'inception_v3', 'mnasnet0_5', 'mnasnet0_75', 'mnasnet1_0', 'mnasnet1_3', 'mobilenet_v2', 'resnet101', 'resnet152', 'resnet18', 'resnet34', 'resnet50', 'resnext101_32x8d', 'resnext50_32x4d', 'shufflenet_v2_x0_5', 'shufflenet_v2_x1_0', 'squeezenet1_0', 'squeezenet1_1', 'vgg11', 'vgg11_bn', 'vgg13', 'vgg13_bn', 'vgg16', 'vgg16_bn', 'vgg19', 'vgg19_bn', 'wide_resnet101_2', 'wide_resnet50_2']
```

```
alexnet = torch.hub.load('pytorch/vision', 'alexnet', pretrained=True)
print(alexnet)
```

打印出模型的信息，通过之前对于卷积神经网络的学习，看看 AlexNet 模型是怎么搭建起来的。
若 pretrained = True，则下载官方在 ImageNet 数据集上预训练好的模型参数。

```
AlexNet(
  (features): Sequential(
    (0): Conv2d(3, 64, kernel_size=(11, 11), stride=(4, 4), padding=(2, 2))
    (1): ReLU(inplace=True)
    (2): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
    (3): Conv2d(64, 192, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
    (4): ReLU(inplace=True)
    (5): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
    (6): Conv2d(192, 384, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (7): ReLU(inplace=True)
    (8): Conv2d(384, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (9): ReLU(inplace=True)
    (10): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (11): ReLU(inplace=True)
    (12): MaxPool2d(kernel_size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (avgpool): AdaptiveAvgPool2d(output_size=(6, 6))
  (classifier): Sequential(
    (0): Dropout(p=0.5, inplace=False)
    (1): Linear(in_features=9216, out_features=4096, bias=True)
    (2): ReLU(inplace=True)
    (3): Dropout(p=0.5, inplace=False)
    (4): Linear(in_features=4096, out_features=4096, bias=True)
    (5): ReLU(inplace=True)
    (6): Linear(in_features=4096, out_features=1000, bias=True)
  )
)
```

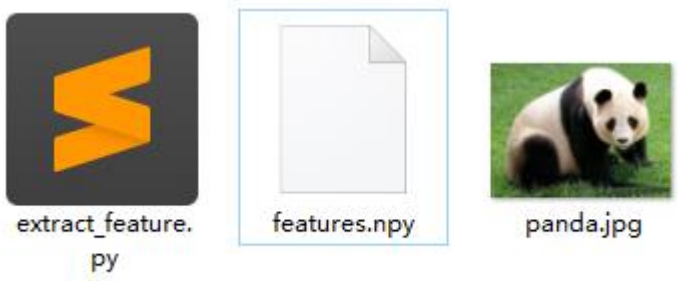
同学们可以用这种方法查看其他的模型都是如何搭建起来的，比如我们之后用到的 ResNet50 模型。

使用 PyTorch 提取图像特征

请大家参考给出的示例代码 extract_features.py，学习如何用 PyTorch 提取示例图像 panda.jpg 的特征。直接运行程序得到如下结果。

```
root@cad54f9508a7:/workspaces/hello_world/data# python extract_feature.py
Load model: ResNet50
Using cache found in /root/.cache/torch/hub/pytorch_vision_master
Prepare image data!
Extract features!
Time for extracting features: 0.27
Save features!
```

注：第一次运行程序，会自动下载预训练好的模型，请保持网络的通畅。在 CPU 上，用 ResNet50 模型提取一张图片大概要 0.3s，不同的电脑配置可能有所差异。



导入ResNet50模型，并加载预训练参数，定义处理图片的归一化操作和预处理方式。

```
print('Load model: ResNet50')
model = torch.hub.load('pytorch/vision', 'resnet50',
pretrained=True)
normalize = transforms.Normalize(mean=[0.485, 0.456, 0.406],
std=[0.229, 0.224, 0.225])
trans = transforms.Compose([ transforms.Resize(256),
transforms.CenterCrop(224), transforms.ToTensor(), normalize])
```

读入图片，并使用之前定义好的处理方式(trans)处理图片，为送入模型提特征做准备。

```
print('Prepare image data!')
test_image = default_loader('panda.jpg')
input_image = trans(test_image)
input_image = torch.unsqueeze(input_image, 0)
```

使用 PyTorch 提取图像特征

正常调用模型 model(input_image)，会返回 ResNet50 模型在 ImageNet 数据集上的最终分类结果。通常来讲，我们可以把模型最终分类的前一层当作模型学习到的图像特征，并用这种特征来完成我们的图像检索任务。

所以我们需要重新定义一个函数features()来提取模型倒数第二层的输出结果。

```
def features(x):
    x = model.conv1(x)
    x = model.bn1(x)
    x = model.relu(x)
    x = model.maxpool(x)
    x = model.layer1(x)
    x = model.layer2(x)
    x = model.layer3(x)
    x = model.layer4(x)
    x = model.avgpool(x)
    return x
```

请同学们结合最开始的模型信息思考函数 features() 为什么要这么写。如果换成其他的模型又该怎么完成 features() 这个函数。

```
print('Extract features!')
start = time.time()
image_feature = features(input_image)
image_feature = image_feature.detach().numpy()
print('Time for extracting features:
{:.2f}'.format(time.time()-start))

print('Save features!')
np.save('features.npy', image_feature)
```

调用函数 features() 完成示例图像 panda.jpg 特征的提取，并将特征保存下来。

图像检索

向量A　　向量B　　向量C

保存得到的特征为一个向量，如何计算向量之间的相似程度呢？

方法一：首先将向量归一化，计算向量之间的欧氏距离，通过距离的远近来反映向量之间的相似程度。

方法二：通过计算向量之间的夹角，夹角越大则越不相似，反之越相似。

练习

- 本次实验要求构建一个不小于 500 张的图像库，用深度学习模型提取图像的特征，并使用上一页提到的方法，计算图片之间的相似度。
- 请使用一些不在图像库中的图片进行测试，完成以图搜图的检索任务。
- 报告中要求给出检索的结果，与被检索图片 Top5 相似的图片和排序，说明排序的方法，给出排序的得分情况。

拓展思考

- 我们在 Lab 7 中学习了 LSH 加速检索，能否使用本次实验提取的 CNN 特征进行 LSH 检索？