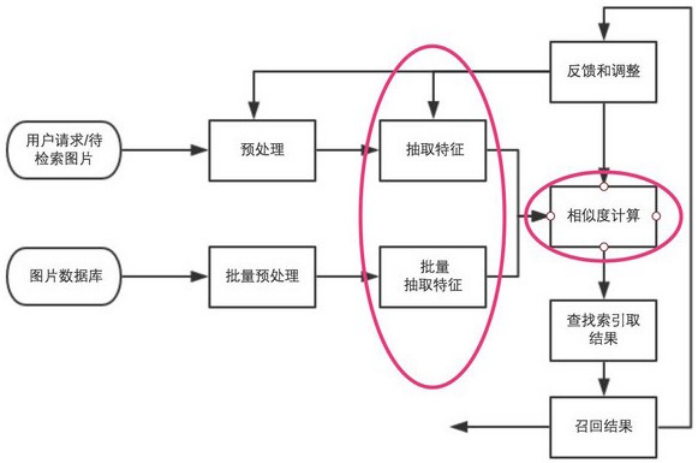
似水流年

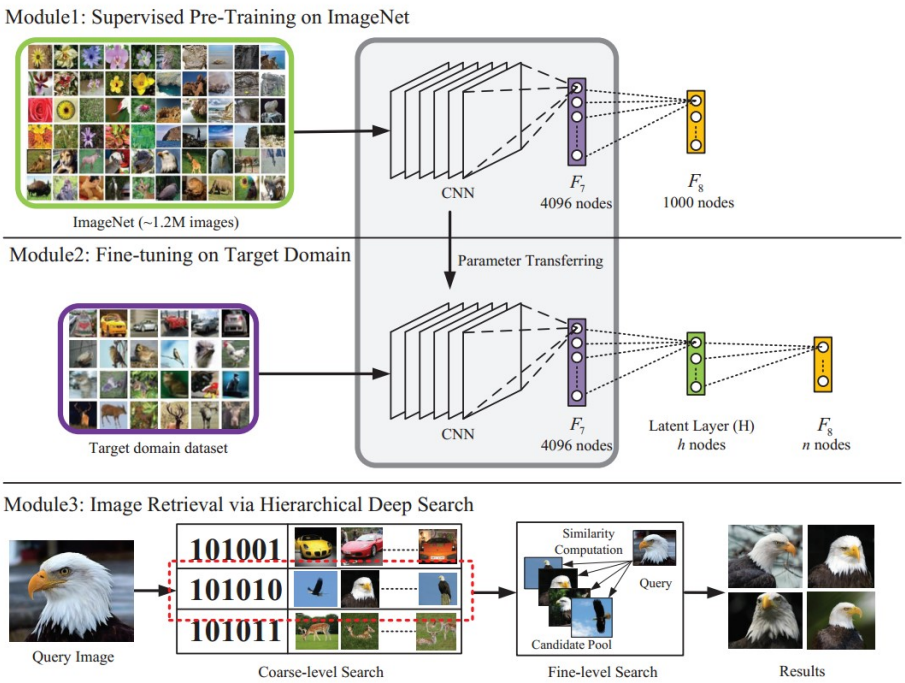
基于VGG-16的海量图像检索系统（以图搜图升级版）

检索系统原理:

图像检索过程简单说来就是对图片数据库的每张图片抽取特征(一般形式为特征向量)，存储于数据库中，对于待检索图片，抽取同样的特征向量，然后并对该向量和数据库中向量的距离（相似度计算），找出最接近的一些特征向量，其对应的图片即为检索结果。[1]



【论文解析概述】下图为ImageNet比赛中使用的卷积神经网络；中间图为调整后，在第7层和output层之间添加隐层(假设为128个神经元)后的卷积神经网络，我们将复用ImageNet中得到最终模型的前7层权重做fine-tuning，得到第7层、8层和output层之间的权重。下方图为实际检索过程，对于所有的图片做卷积神经网络前向运算得到第7层4096维特征向量和第8层128维输出(设定阈值0.5之后可以转成01二值检索向量)，对于待检索的图片，同样得到4096维特征向量和128维01二值检索向量，在数据库中查找二值检索向量对应『桶』内图片，比对4096维特征向量之间距离，做重拍即得到最终结果。图上的检索例子比较直观，对于待检索的"鹰"图像，算得二值检索向量为101010，取出桶内图片(可以看到基本也都为鹰)，比对4096维特征向量之间距离，重新排序拿得到最后的检索结果。



原理部分详见论文，以下是代码实现：

开发环境：

```
# windows 10
# tensorflow-gpu 1.8 + keras
# python 3.6
```

执行示例：

```
# 对database文件夹内图片进行特征提取，建立索引文件featureCNN.h5
python index.py -database database -index featureCNN.h5

# 使用database文件夹内001_accordion_image_0001.jpg作为测试图片，在database内以featureCNN.h5进行近似图片查找，并显示最近似的3张图片
python query_online.py -query database/001_accordion_image_0001.jpg -index featureCNN.h5 -result database
```

1、抽取特征：extract_cnn_vgg16_keras.py

```
# -*- coding: utf-8 -*-

import numpy as np
from numpy import linalg as LA

from keras.applications.vgg16 import VGG16
from keras.preprocessing import image
from keras.applications.vgg16 import preprocess_input

class VGGNet:
    def __init__(self):
        # weights: 'imagenet'
        # pooling: 'max' or 'avg'
        # input_shape: (width, height, 3), width and height should >= 48
        self.input_shape = (224, 224, 3)
        self.weight = 'imagenet'
        self.pooling = 'max'
        self.model = VGG16(weights = self.weight, input_shape = (self.input_shape[0], self.input_shape[1], self.input_shape[2]), pooling = self.pooling, include_top = False)
        self.model.predict(np.zeros((1, 224, 224 , 3)))
```

java(1)
更多

随笔档案
2019年10月(6)
2019年9月(2)
2019年8月(8)
2019年7月(1)
2019年6月(14)
2019年5月(17)
2019年3月(1)
2019年1月(4)
2018年10月(1)
2018年9月(2)

文章分类
大数据(1)
机器学习(1)
数据结构(1)

最新评论
1. Re:hadoop启动集群置
假如 A 要登陆 B在A上成密钥对ssh-keygen (车即可)%%再将A自己到B的授权列表文件auth ssh-copy-id ...
2. Re:基于VGG-16的海（以图搜图升级版）
我按照你的方法提取的特怎么提取4096维的呢

```
    '''
    Use vgg16 model to extract features
    Output normalized feature vector
    '''
    def extract_feat(self, img_path):
        img = image.load_img(img_path, target_size=(self.input_shape[0], self.input_shape[1]))
        img = image.img_to_array(img)
        img = np.expand_dims(img, axis=0)
        img = preprocess_input(img)
        feat = self.model.predict(img)
        norm_feat = feat[0]/LA.norm(feat[0])
        return norm_feat
```

2、存储索引：index.py

```
# -*- coding: utf-8 -*-
import os
import h5py
import numpy as np
import argparse

from extract_cnn_vgg16_keras import VGGNet


ap = argparse.ArgumentParser()
ap.add_argument("-database", required = True,
    help = "Path to database which contains images to be indexed")
ap.add_argument("-index", required = True,
    help = "Name of index file")
args = vars(ap.parse_args())

'''
 Returns a list of filenames for all jpg images in a directory.
'''
def get_imlist(path):
    return [os.path.join(path,f) for f in os.listdir(path) if f.endswith('.jpg')]

'''
 Extract features and index the images
'''
if __name__ == "__main__":

    db = args["database"]
    img_list = get_imlist(db)

    print ("--------------------------------------------------")
    print ("         feature extraction starts")
    print ("--------------------------------------------------")

    feats = []
    names = []

    model = VGGNet()
    for i, img_path in enumerate(img_list):
        norm_feat = model.extract_feat(img_path)
        img_name = os.path.split(img_path)[1]
        feats.append(norm_feat)
        names.append(img_name.encode())
        print ("extracting feature from image No. %d , %d images in total" %((i+1), len(img_list)))

    feats = np.array(feats)
    # directory for storing extracted features
    output = args["index"]

    print ("--------------------------------------------------")
    print ("      writing feature extraction results ...")
    print ("--------------------------------------------------")

    h5f = h5py.File(output, 'w')
    h5f.create_dataset('dataset_1', data = feats)
    h5f.create_dataset('dataset_2', data = names)
    h5f.close()
```

3、在线搜索部分query_online.py：

```
# -*- coding: utf-8 -*-
from extract_cnn_vgg16_keras import VGGNet

import numpy as np
import h5py

import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import argparse


ap = argparse.ArgumentParser()
ap.add_argument("-query", required = True,
    help = "Path to query which contains image to be queried")
ap.add_argument("-index", required = True,
    help = "Path to index")
ap.add_argument("-result", required = True,
    help = "Path for output retrieved images")
args = vars(ap.parse_args())


# read in indexed images' feature vectors and corresponding image names
h5f = h5py.File(args["index"],'r')
feats = h5f['dataset_1'][:]
imgNames = h5f['dataset_2'][:]
h5f.close()

print ("--------------------------------------------------")
print ("               searching starts")
print ("--------------------------------------------------")

# read and show query image
queryDir = args["query"]
queryImg = mpimg.imread(queryDir)
plt.title("Query Image")
plt.imshow(queryImg)
plt.show()

# init VGGNet16 model
model = VGGNet()

# extract query image's feature, compute simlarity score and sort
queryVec = model.extract_feat(queryDir)
scores = np.dot(queryVec, feats.T)
rank_ID = np.argsort(scores)[::-1]
rank_score = scores[rank_ID]
#print rank_ID
#print rank_score


# number of top retrieved images to show
maxres = 3
imlist = [imgNames[index] for i,index in enumerate(rank_ID[0:maxres])]
print ("top %d images in order are: " %maxres, imlist)


# show top #maxres retrieved result one by one
for i,im in enumerate(imlist):
    image = mpimg.imread(args["result"]+"/"+str(im,encoding='utf-8'))
    plt.title("search output %d" %(i+1))
    plt.imshow(image)
    plt.show()
```

参考及引用：

利用VGG16提取特征：https://keras-cn.readthedocs.io/en/latest/other/application/

图片检索方法：https://github.com/willard-yuan

论文推荐：https://github.com/willard-yuan/awesome-cbir-papers

论文：http://www.iis.sinica.edu.tw/~kevinlin311.tw/cvprw15.pdf

[1]：https://blog.csdn.net/han_xiaoyang/article/details/50856583

posted @ 2018-09-07 23:43 消失的白桦林 阅读(2834) 评论(1) 编辑 收藏

评论列表

#1楼 2019-01-23 19:18 剑翎

我按照你的方法提取的特征是512维的。怎么提取4096维的呢

支持(1) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册， 访问 网站首页。