Caffe 提取任意层特征并进行可视化 - 普兒

时间 2014-09-11 20:51:53 博客园-所有随笔区原文 http://www.cnblogs.com/platero/p/3967208.html

原图

conv1 层可视化结果 （96 个 filter 得到的结果）

数据模型与准备

安装好 Caffe 后，在 examples/images 文件夹下有两张示例图像，本文即在这两张图像上，用 Caffe 提供的预训练模型，进行特征提取，并进行可视化。

1. 进入 caffe 根目录，创建临时文件夹，用于存放所需要的临时文件

```
mkdir examples/_temp
```

2. 根据 examples/images 文件夹中的图片，创建包含图像列表的 txt 文件，并添加标签（0）

```
find `pwd`/examples/images -type f -exec echo {} \; > examples/_temp/temp.txt

sed "s/$/ 0/" examples/_temp/temp.txt > examples/_temp/file_list.txt
```

3. 执行下列脚本，下载 imagenet12 图像均值文件，在后面的网络结构定义 prototxt 文件中，需要用到该文件 （data/ilsvrc212/imagenet_mean.binaryproto）

```
data/ilsvrc12/get_ilsvrc_aux.sh
```

4. 将网络定义 prototxt 文件复制到_temp 文件夹下

```
cp examples/feature_extraction/imagenet_val.prototxt examples/_temp
```

提取特征

1. 创建 src/youname/ 文件夹， 存放我们自己的脚本

```
mkdir src/yourname
```

2. caffe 的 extract_features 将提取出的图像特征存为 leveldb 格式， 为了方便观察特征，我们将利用下列两个 python 脚本将图像转化为 matlab 的.mat 格式 （请先安装 caffe 的 python 依赖库）

feat_helper_pb2.py

```
# Generated by the protocol buffer compiler.  DO NOT EDIT!



from google.protobuf import descriptor

from google.protobuf import message

from google.protobuf import reflection

from google.protobuf import descriptor_pb2

# @@protoc_insertion_point(imports)






DESCRIPTOR = descriptor.FileDescriptor(

  name='datum.proto',

  package='feat_extract',

  serialized_pb='\n\x0b\x64\x61tum.proto\x12\x0c\x66\x65\x61t_extract\"i\n\x05\x44\x61tum\x12\x10\n\x08\x63hannels\x18\x01 \x01(\x05\x12\x0e\n\x06height\x18\x02 \x01(\x05\x12\r\n\x05width\x18\x03 \x01(\x05\x12\x0c\n\x04\x64\x61ta\x18\x04 \x01(\x0c\x12\r\n\x05label\x18\x05 \x01(\x05\x12\x12\n\nfloat_data\x18\x06 \x03(\x02')
_DATUM = descriptor.Descriptor(
  name='Datum',
  full_name='feat_extract.Datum',
  filename=None,
  file=DESCRIPTOR,
  containing_type=None,
  fields=[
    descriptor.FieldDescriptor(
      name='channels', full_name='feat_extract.Datum.channels', index=
0,
      number=1, type=5, cpp_type=1, label=1,
      has_default_value=False, default_value=0,
      message_type=None, enum_type=None, containing_type=None,
      is_extension=False, extension_scope=None,
      options=None),
    descriptor.FieldDescriptor(
      name='height', full_name='feat_extract.Datum.height', index=1,
      number=2, type=5, cpp_type=1, label=1,
      has_default_value=False, default_value=0,
      message_type=None, enum_type=None, containing_type=None,
      is_extension=False, extension_scope=None,
      options=None),
    descriptor.FieldDescriptor(
      name='width', full_name='feat_extract.Datum.width', index=2,
      number=3, type=5, cpp_type=1, label=1,
      has_default_value=False, default_value=0,
      message_type=None, enum_type=None, containing_type=None,
      is_extension=False, extension_scope=None,
      options=None),
    descriptor.FieldDescriptor(
      name='data', full_name='feat_extract.Datum.data', index=3,
      number=4, type=12, cpp_type=9, label=1,
      has_default_value=False, default_value="",
      message_type=None, enum_type=None, containing_type=None,
      is_extension=False, extension_scope=None,
      options=None),
    descriptor.FieldDescriptor(
      name='label', full_name='feat_extract.Datum.label', index=4,
      number=5, type=5, cpp_type=1, label=1,
      has_default_value=False, default_value=0,
      message_type=None, enum_type=None, containing_type=None,
      is_extension=False, extension_scope=None,
      options=None),
    descriptor.FieldDescriptor(
      name='float_data', full_name='feat_extract.Datum.float_data', in
dex=5,
      number=6, type=2, cpp_type=6, label=3,
      has_default_value=False, default_value=[],
      message_type=None, enum_type=None, containing_type=None,
      is_extension=False, extension_scope=None,
      options=None),
  ],
  extensions=[
  ],
  nested_types=[],
  enum_types=[
  ],
  options=None,
  is_extendable=False,
  extension_ranges=[],
  serialized_start=29,
  serialized_end=134,
)


DESCRIPTOR.message_types_by_name['Datum'] = _DATUM


class Datum(message.Message):
  __metaclass__ = reflection.GeneratedProtocolMessageType
  DESCRIPTOR = _DATUM
  # @@protoc_insertion_point(class_scope:feat_extract.Datum)


# @@protoc_insertion_point(module_scope)
import leveldb
import feat_helper_pb2
import numpy as np
import scipy.io as sio
import time


def main(argv):
  leveldb_name = sys.argv[1]
  print "%s" % sys.argv[1]
  batch_num = int(sys.argv[2]);
  batch_size = int(sys.argv[3]);
  window_num = batch_num*batch_size;


  start = time.time()
  if 'db' not in locals().keys():
    db = leveldb.LevelDB(leveldb_name)
    datum = feat_helper_pb2.Datum()

  ft = np.zeros((window_num, int(sys.argv[4])))
  for im_idx in range(window_num):
    datum.ParseFromString(db.Get('%d' %(im_idx)))
    ft[im_idx, :] = datum.float_data

  print 'time 1: %f' %(time.time() - start)
  sio.savemat(sys.argv[5], {'feats':ft})
  print 'time 2: %f' %(time.time() - start)
  print 'done!'

  #leveldb.DestroyDB(leveldb_name)

if __name__ == '__main__':
  import sys
  main(sys.argv)
```

3. 创建脚本文件 extract_feature.sh， 并执行，将在 examples/_temp 文件夹下得到 leveldb 文件（features_conv1）和.mat 文件（features.mat）

```
#!/usr/bin/env sh

# args for EXTRACT_FEATURE

TOOL=../../build/tools

MODEL=../../examples/imagenet/caffe_reference_imagenet_model #下载得到的caffe model

PROTOTXT=../../examples/_temp/imagenet_val.prototxt # 网络定义

LAYER=conv1 # 提取层的名字，如提取fc7 等

LEVELDB=../../examples/_temp/features_conv1 # 保存的leveldb 路径

BATCHSIZE=10



# args for LEVELDB to MAT

DIM=290400 # 需要手工计算feature 长度

OUT=../../examples/_temp/features.mat #.mat 文件保存路径

BATCHNUM=1 # 有多少哥batch， 本例只有两张图， 所以只有一个batch



$TOOL/extract_features.bin  $MODEL $PROTOTXT $LAYER $LEVELDB $BATCHSIZE

python leveldb2mat.py $LEVELDB $BATCHNUM  $BATCHSIZE $DIM $OUT
```

4. 得到.mat 文件后，需要对其进行可视化，这里用了 UFLDL 里的 display_network 函数，由于可视化出来结果进行了翻转，因此对原代码的 67, 69, 83, 85 行进行了修改

display_network.m 存放在 src/yourname 文件夹下

```
function [h, array] = display_network(A, opt_normalize, opt_graycolor, cols, opt_colmajor)

% This function visualizes filters in matrix A. Each column of A is a

% filter. We will reshape each column into a square image and visualizes

% on each cell of the visualization panel.

% All other parameters are optional, usually you do not need to worry

% about it.

% opt_normalize: whether we need to normalize the filter so that all of

% them can have similar contrast. Default value is true.

% opt_graycolor: whether we use gray as the heat map. Default is true.

% cols: how many columns are there in the display. Default value is the

% squareroot of the number of columns in A.

% opt_colmajor: you can switch convention to row major for A. In that

% case, each row of A is a filter. Default value is false.

warning off all
```

```
if ~exist('opt_normalize', 'var') || isempty(opt_normalize)
  opt_normalize= true;
end

if ~exist('opt_graycolor', 'var') || isempty(opt_graycolor)
  opt_graycolor= true;
end

if ~exist('opt_colmajor', 'var') || isempty(opt_colmajor)
  opt_colmajor = false;
end

% rescale
A = A - mean(A(:));

if opt_graycolor, colormap(gray); end
```

```
% compute rows, cols
[L M]=size(A);
sz=sqrt(L);
buf=1;
if ~exist('cols', 'var')
  if floor(sqrt(M))^2 ~= M
    n=ceil(sqrt(M));
    while mod(M, n)~=0 && n<1.2*sqrt(M), n=n+1; end
    m=ceil(M/n);
  else
    n=sqrt(M);
    m=n;
  end
else
  n = cols;
  m = ceil(M/n);
end
```

```
array=-ones(buf+m*(sz+buf),buf+n*(sz+buf));

if ~opt_graycolor
  array = 0.1.* array;
end


if ~opt_colmajor
  k=1;
  for i=1:m
    for j=1:n
      if k>M,
        continue;
      end
      clim=max(abs(A(:,k)));
      if opt_normalize
        array(buf+(i-1)*(sz+buf)+(1:sz),buf+(j-1)*(sz+buf)+(1:sz))=reshape(A(:,k),sz,sz)'/clim;
      else
```

```
        array(buf+(i-1)*(sz+buf)+(1:sz),buf+(j-1)*(sz+buf)+(1:sz))=reshape(A(:,k),sz,sz)'/max(abs(A(:)));
      end
      k=k+1;
    end
  end
else
  k=1;
  for j=1:n
    for i=1:m
      if k>M,
        continue;
      end
      clim=max(abs(A(:,k)));
      if opt_normalize
        array(buf+(i-1)*(sz+buf)+(1:sz),buf+(j-1)*(sz+buf)+(1:sz))=reshape(A(:,k),sz,sz)'/clim;
      else
        array(buf+(i-1)*(sz+buf)+(1:sz),buf+(j-1)*(sz+buf)+(1:sz))=reshape(A(:,k),sz,sz)';
      end

      k=k+1;

    end

  end

end



if opt_graycolor

  h=imagesc(array,'EraseMode','none',[-1 1]);

else

  h=imagesc(array,'EraseMode','none',[-1 1]);

end

axis image off



drawnow;



warning on all
```

5. 调用 display_network 以及提取到的 feature 进行可视化：

在 examples/_temp/ 下创建如下 matlab 脚本， 并执行

```
addpath(genpath('../../src/wyang'));
```

```
nsample     = 3;
num_output  = 96;

load features.mat
width = size(feats, 2);
nmap  = width / num_output;

for i = 1:nsample
    feat = feats(i, :);
    feat = reshape(feat, [nmap num_output]);
    figure('name', sprintf('image #%d', i));
    display_network(feat);
end
```