fujisan

use Keras pre-trained VGG16 acc 98%

last run a year ago · Python notebook · 19413 views
using data from Invasive Species Monitoring · Public

Notebook

use Keras pre-trained VGG16

this is my first notebook.

pre-trained VGG16 is quickly and good performance.

I learned from official Keras blog tutorial Building powerful image classification models using very little data (https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html)

resize train data and test data


```
import numpy as np
import pandas as pd
import cv2
import math
from glob import glob
import os

master = pd.read_csv("../input/train_labels.csv")
master.head()
```

Out[1]:

	name	invasive
0	1	0
1	2	0
2	3	1
3	4	0
4	5	1

In [2]:

```
img_path = "../input/train/"

y = []
file_paths = []
for i in range(len(master)):
    file_paths.append( img_path + str(master.ix[i][0]) +'.jpg' )
    y.append(master.ix[i][1])
y = np.array(y)
```

In [3]:

```
#image reseize & centering & crop

def centering_image(img):
    size = [256,256]
```

```
    img_size = img.shape[:2]

    # centering
    row = (size[1] - img_size[0]) // 2
    col = (size[0] - img_size[1]) // 2
    resized = np.zeros(list(size) + [img.shape[2]], dtype=np.uint8)
    resized[row:(row + img.shape[0]), col:(col + img.shape[1])] = img

    return resized


x = []
for i, file_path in enumerate(file_paths):
    #read image
    img = cv2.imread(file_path)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    #resize
    if(img.shape[0] > img.shape[1]):
        tile_size = (int(img.shape[1]*256/img.shape[0]),256)
    else:
        tile_size = (256, int(img.shape[0]*256/img.shape[1]))

    #centering
    img = centering_image(cv2.resize(img, dsize=tile_size))

    #out put 224*224px
    img = img[16:240, 16:240]
    x.append(img)

x = np.array(x)
```

In [4]:

```
sample_submission = pd.read_csv("../input/sample_submission.csv")
img_path = "../input/test/"

test_names = []
file_paths = []

for i in range(len(sample_submission)):
    test_names.append(sample_submission.ix[i][0])
    file_paths.append( img_path + str(int(sample_submission.ix[i][0])) +'.jpg' )

test_names = np.array(test_names)
```

In [5]:

```
test_images = []
for file_path in file_paths:
    #read image
    img = cv2.imread(file_path)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    #resize
    if(img.shape[0] > img.shape[1]):
```

```
        tile_size = (int(img.shape[1]*256/img.shape[0]),256)
    else:
        tile_size = (256, int(img.shape[0]*256/img.shape[1]))

    #centering
    img = centering_image(cv2.resize(img, dsize=tile_size))

    #out put 224*224px
    img = img[16:240, 16:240]
    test_images.append(img)

    path, ext = os.path.splitext( os.path.basename(file_paths[0]) )

test_images = np.array(test_images)
```

save numpy array.

Usually I separate code, data format and CNN.

In [6]:
```
#np.savez('224.npz', x=x, y=y, test_images=test_images, test_names=test_names)
```

split train data and validation data

In [7]:
```
data_num = len(y)
random_index = np.random.permutation(data_num)

x_shuffle = []
y_shuffle = []
for i in range(data_num):
    x_shuffle.append(x[random_index[i]])
    y_shuffle.append(y[random_index[i]])

x = np.array(x_shuffle)
y = np.array(y_shuffle)
```

In [8]:
```
val_split_num = int(round(0.2*len(y)))
x_train = x[val_split_num:]
y_train = y[val_split_num:]
x_test = x[:val_split_num]
y_test = y[:val_split_num]

print('x_train', x_train.shape)
print('y_train', y_train.shape)
print('x_test', x_test.shape)
print('y_test', y_test.shape)
```

```
x_train (1836, 224, 224, 3)
y_train (1836,)
```

```
x_test (459, 224, 224, 3)
y_test (459,)
```

In [9]:
```
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
```

use Keras pre-trained VGG16

but kaggle karnel is not run

In [10]:
```
from keras.models import Sequential, Model, load_model
from keras import applications
from keras import optimizers
from keras.layers import Dropout, Flatten, Dense

img_rows, img_cols, img_channel = 224, 224, 3

base_model = applications.VGG16(weights='imagenet', include_top=False, input_shape=(img_rows, img_cols, img_channel))
```

```
Using TensorFlow backend.

Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5

---------------------------------------------------------------------------
gaierror                                  Traceback (most recent call last)
/opt/conda/lib/python3.6/urllib/request.py in do_open(self, http_class, req, **http_conn_args)
   1317                 h.request(req.get_method(), req.selector, req.data, headers,
-> 1318                           encode_chunked=req.has_header('Transfer-encoding'))
   1319             except OSError as err: # timeout error

/opt/conda/lib/python3.6/http/client.py in request(self, method, url, body, headers, encode_chunked)
   1238         """Send a complete request to the server."""
-> 1239         self._send_request(method, url, body, headers, encode_chunked)
   1240

/opt/conda/lib/python3.6/http/client.py in _send_request(self, method, url, body, headers, encode_chunked)
   1284             body = _encode(body, 'body')
-> 1285         self.endheaders(body, encode_chunked=encode_chunked)
   1286

/opt/conda/lib/python3.6/http/client.py in endheaders(self, message_body, encode_chunked)
   1233             raise CannotSendHeader()
-> 1234         self._send_output(message_body, encode_chunked=encode_chunked)
   1235
```

```
/opt/conda/lib/python3.6/http/client.py in _send_output(self, message_body, encode_chunked)
   1025         del self._buffer[:]
-> 1026         self.send(msg)
   1027

/opt/conda/lib/python3.6/http/client.py in send(self, data)
    963             if self.auto_open:
--> 964                 self.connect()
    965             else:

/opt/conda/lib/python3.6/http/client.py in connect(self)
   1391
-> 1392             super().connect()
   1393

/opt/conda/lib/python3.6/http/client.py in connect(self)
    935         self.sock = self._create_connection(
--> 936             (self.host,self.port), self.timeout, self.source_address)
    937         self.sock.setsockopt(socket.IPPROTO_TCP, socket.TCP_NODELAY, 1)

/opt/conda/lib/python3.6/socket.py in create_connection(address, timeout, source_address)
    703     err = None
--> 704     for res in getaddrinfo(host, port, 0, SOCK_STREAM):
    705         af, socktype, proto, canonname, sa = res

/opt/conda/lib/python3.6/socket.py in getaddrinfo(host, port, family, type, proto, flags)
    742     addrlist = []
--> 743     for res in _socket.getaddrinfo(host, port, family, type, proto, flags):
    744         af, socktype, proto, canonname, sa = res

gaierror: [Errno -2] Name or service not known

During handling of the above exception, another exception occurred:

URLError                                  Traceback (most recent call last)
/opt/conda/lib/python3.6/site-packages/Keras-2.0.4-py3.6.egg/keras/utils/data_utils.py in get_
file(fname, origin, untar, md5_hash, file_hash, cache_subdir, hash_algorithm, extract, archive
_format, cache_dir)
    200                 urlretrieve(origin, fpath,
--> 201                             functools.partial(dl_progress, progbar=progbar))
    202             except URLError as e:

/opt/conda/lib/python3.6/urllib/request.py in urlretrieve(url, filename, reporthook, data)
    247
--> 248     with contextlib.closing(urlopen(url, data)) as fp:
    249         headers = fp.info()

/opt/conda/lib/python3.6/urllib/request.py in urlopen(url, data, timeout, cafile, capath, cade
fault, context)
    222         opener = _opener
--> 223     return opener.open(url, data, timeout)
    224

/opt/conda/lib/python3.6/urllib/request.py in open(self, fullurl, data, timeout)
```

```
    525
--> 526         response = self._open(req, data)
    527

/opt/conda/lib/python3.6/urllib/request.py in _open(self, req, data)
    543         result = self._call_chain(self.handle_open, protocol, protocol +
--> 544                                   '_open', req)
    545         if result:

/opt/conda/lib/python3.6/urllib/request.py in _call_chain(self, chain, kind, meth_name, *args)
    503             func = getattr(handler, meth_name)
--> 504             result = func(*args)
    505             if result is not None:

/opt/conda/lib/python3.6/urllib/request.py in https_open(self, req)
   1360             return self.do_open(http.client.HTTPSConnection, req,
-> 1361                 context=self._context, check_hostname=self._check_hostname)
   1362

/opt/conda/lib/python3.6/urllib/request.py in do_open(self, http_class, req, **http_conn_args)
   1319             except OSError as err: # timeout error
-> 1320                 raise URLError(err)
   1321             r = h.getresponse()

URLError: <urlopen error [Errno -2] Name or service not known>

During handling of the above exception, another exception occurred:

Exception                                 Traceback (most recent call last)
<ipython-input-10-0287e0db5c96> in <module>()
      6 img_rows, img_cols, img_channel = 224, 224, 3
      7
----> 8 base_model = applications.VGG16(weights='imagenet', include_top=False, input_shape=(im
g_rows, img_cols, img_channel))

/opt/conda/lib/python3.6/site-packages/Keras-2.0.4-py3.6.egg/keras/applications/vgg16.py in VG
G16(include_top, weights, input_tensor, input_shape, pooling, classes)
    166             weights_path = get_file('vgg16_weights_tf_dim_ordering_tf_kernels_notop.h
5',
    167                                     WEIGHTS_PATH_NO_TOP,
--> 168                                     cache_subdir='models')
    169         model.load_weights(weights_path)
    170         if K.backend() == 'theano':

/opt/conda/lib/python3.6/site-packages/Keras-2.0.4-py3.6.egg/keras/utils/data_utils.py in get_
file(fname, origin, untar, md5_hash, file_hash, cache_subdir, hash_algorithm, extract, archive
_format, cache_dir)
    201                                 functools.partial(dl_progress, progbar=progbar))
    202             except URLError as e:
--> 203                 raise Exception(error_msg.format(origin, e.errno, e.reason))
    204             except HTTPError as e:
    205                 raise Exception(error_msg.format(origin, e.code, e.msg))

Exception: URL fetch failure on https://github.com/fchollet/deep-learning-models/releases/down
load/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5: None -- [Errno -2] Name or servic
e not known
```

e not known

In [11]:
```
add_model = Sequential()
add_model.add(Flatten(input_shape=base_model.output_shape[1:]))
add_model.add(Dense(256, activation='relu'))
add_model.add(Dense(1, activation='sigmoid'))

model = Model(inputs=base_model.input, outputs=add_model(base_model.output))
model.compile(loss='binary_crossentropy', optimizer=optimizers.SGD(lr=1e-4, momentum=0.9),
              metrics=['accuracy'])

model.summary()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-11-86cf87e2cb26> in <module>()
      1 add_model = Sequential()
----> 2 add_model.add(Flatten(input_shape=base_model.output_shape[1:]))
      3 add_model.add(Dense(256, activation='relu'))
      4 add_model.add(Dense(1, activation='sigmoid'))
      5

NameError: name 'base_model' is not defined
```

In [12]:
```
from keras.preprocessing.image import ImageDataGenerator
from keras.callbacks import ModelCheckpoint

batch_size = 32
epochs = 50

train_datagen = ImageDataGenerator(
        rotation_range=30,
        width_shift_range=0.1,
        height_shift_range=0.1,
        horizontal_flip=True)
train_datagen.fit(x_train)


history = model.fit_generator(
    train_datagen.flow(x_train, y_train, batch_size=batch_size),
    steps_per_epoch=x_train.shape[0] // batch_size,
    epochs=epochs,
    validation_data=(x_test, y_test),
    callbacks=[ModelCheckpoint('VGG16-transferlearning.model', monitor='val_acc', save_best_on
ly=True)]
)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-12-2e0a31593721> in <module>()
     13
     14
```

```
---> 15 history = model.fit_generator(
     16     train_datagen.flow(x_train, y_train, batch_size=batch_size),
     17     steps_per_epoch=x_train.shape[0] // batch_size,

NameError: name 'model' is not defined
```

predict test data

In [13]:
```
test_images = test_images.astype('float32')
test_images /= 255
```

In [14]:
```
predictions = model.predict(test_images)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-14-4f3b02264dd8> in <module>()
----> 1 predictions = model.predict(test_images)

NameError: name 'model' is not defined
```

In [15]:
```
sample_submission = pd.read_csv("../input/sample_submission.csv")

for i, name in enumerate(test_names):
    sample_submission.loc[sample_submission['name'] == name, 'invasive'] = predictions[i]

sample_submission.to_csv("submit.csv", index=False)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-15-8d1f4d5c4cae> in <module>()
      2
      3 for i, name in enumerate(test_names):
----> 4     sample_submission.loc[sample_submission['name'] == name, 'invasive'] = predictions
[i]
      5
      6 sample_submission.to_csv("submit.csv", index=False)

NameError: name 'predictions' is not defined
```

Please sign in to leave a comment.

Yassine Ghouzam • Posted on Latest Version • 8 months ago • Options 7

You didn't try to finetune the vgg16 model instead of retrain all the weights ?

You can freeze some layers by seting the attribut 'trainable' of layers to False:

You can for example freeze all the layers except the your new added layers (add_model) ,the 4 ending layers (maxpool2d +3 conv) with this code :

```
for layer in model.layers[:-5]:
    layer.trainable = False
```

Your vgg16 will use the imagenet weights for the top layers and train only the last 5 layers.

Crequena • Posted on Latest Version • a year ago • Options 7

Nice work! Happy to see that you also found useful F. Chollet 's Keras blog entry!

If you allow me, here is a nice little tip for you to keep playing around and getting further this kernel. We found out that input resolution matter (a lot). The pictures that were not correctly classified had a very small area of *Hydrangea* in them. Increasing the input resolution to 400 x 300 had a quite spectacular increase in accuracy in the "hard to detect set". I can only think that, if the resolution is taken further up, it will have even better results (although benefits might start to be marginal). All of it will happen, of course, at the cost of computational power.

fujisan • Posted on Latest Version • a year ago • Options 0

thanks Crequena.

I will try it.

Ankit Agarwal • Posted on Latest Version • a year ago • Options 0

How would one go about increasing the input resolution ? I mean VGG is supposed to accept only 256/256 images, if we want to use the imagenet pre trained model. Any tips ?

mgenkin • Posted on Latest Version • 10 months ago • Options 0

you could cut the image into squares and take maximum prediction.

chmaxx • Posted on Latest Version • 10 months ago • Options 1

You simply can input any image size to the pretrained network. I tried and it seems to work:
https://www.kaggle.com/chmaxx/finetune-vgg16-0-97-with-minimal-effort

chmaxx • Posted on Latest Version • 10 months ago • Options 0

And thanks, Craquena, for the hint with the bigger image sizes. Very much appreciated.

Robert Wexler • Posted on Latest Version • 10 months ago • Options 0

fujisan, thanks for uploading this Kernel! Crequena, thanks for your suggestion!

I increased the input resolution from 224x224 to 400x400, using the same centering technique included by fujisan in this Kernel, and the accuracy increased from 0.98475 to 0.98943. Training took about 24 hours on 1 CPU, so really not bad at all!

ThyrixYang • Posted on Latest Version • 8 months ago • Options 0

We can change image size because we do not use the fully connected layer in vggnet, which fix the input size. Other layers such as convolution or maxpooling can adjust to different size automatically.

chestnut11 • Posted on Latest Version • 8 months ago • Options 0

can we add a spatial pyramid pooling layer between the fc and pool5?

Nissim Matatov • Posted on Latest Version • 9 months ago • Options 1

Hi ,

On this step :

history = model.fit_generator(train_datagen.flow(x_train, y_train, batch_size=batch_size), steps_per_epoch=x_train.shape[0] // batch_size, epochs=epochs, validation_data=(x_test, y_test), callbacks= [ModelCheckpoint('VGG16-transferlearning.model', monitor='val_acc', save_best_only=True)])

I got error regarding shape inconsistency . Should x_test be transformed with generator ? I think this is the problem

ShubhamShukla • Posted on Latest Version • 10 months ago • Options 1

Helped a lot .... Cheers!!!

Fabian Heinemann • Posted on Latest Version • a year ago • Options 2

Nice notebook. Out of curiosity: why didn't you fix the layers of the pre-trained VGG16 network?

Alex Bath • Posted on Latest Version • a year ago • Options 0

Thanks for sharing!

Luis Bronchal • Posted on Latest Version • a year ago • Options

Great work! What hardware did you use to train this model? How many time did it take?

fujisan • Posted on Latest Version • a year ago • Options 1

I used GTX1080ti, 1 epoch was about 20 seconds.

pre-trained VGG16 is not require much memory and execution time.

mengbo • Posted on Latest Version • 10 months ago • Options 0

thanks for sharing

Marc Spoorendo... • Posted on Latest Version • 10 months ago • Options 0

Could you elaborate on your resizing/centering/cropping section? What objectives are you trying to meet there? My assumption is that it has to do with requirements of the pretrained network, computing time, and maybe other aspects as well. I'm curious about the details.

William Krum... • Posted on Latest Version • 10 months ago • Options 3

Because this is a small dataset, e.i. there are not enough images, they artificially increase the samples size by doing these distortions that do not completely obscure the object being classified. This is done so that features that are independent of scale are used more or rather that there is a greater weighting towards those features that are still common after some artificial variations are applied.

Have you heard of SIFT, scale invariant feature transforms? https://www.wikiwand.com/en/Scale-invariant_feature_transform This is used in photogrammetry to match images bases on features (shapes or patterns of pixels) that are similar and identifiable in images that are taken at different distances. I'm not saying that the neural network is using SIFT, but it looks like a similar process. mmm

Zhizhuo Zhou • Posted on Latest Version • 10 months ago • Options 0

Is training the pre-trained VGG16 resource intensive? I get out of resource error with my 4GB GPU.

Ignacio Vilaplana • Posted on Latest Version • 9 months ago • Options 0

Great code! Thanks

OctavianTuchila • Posted on Latest Version • 9 months ago • Options 0

Great code!

What is this though:

```
Downloading data from https://github.com/fchollet/deep-learning-
models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
```

Why is this downloaded from Github? And how can the code work if this error is triggered:

Exception: URL fetch failure on https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5: None -- [Errno -2] Name or service not known

ThyrixYang • Posted on Latest Version • 8 months ago • Options

0

You can download the pretrained weights from github directly and save it at ~/.keras/models, then it won't need to be download again.

OctavianTuchila • Posted on Latest Version • 9 months ago • Options

0

Great code!

What is this though:

```
Downloading data from https://github.com/fchollet/deep-learning-
models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
```

Why is this downloaded from Github? And how can the code work if this error is triggered:

Exception: URL fetch failure on https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5: None -- [Errno -2] Name or service not known

bminixhofer • Posted on Latest Version • 9 months ago • Options

1

The weights of the model are pretrained (except of the last layer) on imagenet. This code just downloads the weights from the keras repository.

My guess is that it doesn't work because it's run by the kaggle servers which might not support downloading from external resources.

NikitaButakov • Posted on Latest Version • 9 months ago • Options

0

Why are the images centered after resizing? It seems there is some information lost from the boundaries of the image because of the way images are resized, centered, and cropped. Thanks!

sumer • Posted on Latest Version • 9 months ago • Options

0

Amandeep Singh • Posted on Latest Version • 9 months ago • Options 0

Helped a lot.

Harvey • Posted on Latest Version • 8 months ago • Options 0

Great code. Thanks. I am trying if there is a difference using scipy.misc imread compared with CV2.

Xiangyi Yan • Posted on Latest Version • 8 months ago • Options 0

Ogurtsov • Posted on Latest Version • 8 months ago • Options 0

Thank you for the kernel, I have successfully reproduced your results with 1050Ti.

DecentMakeover • Posted on Latest Version • 8 months ago • Options 0

When i try running your model,i keep getting 'load weights' reqiures h5py,even after i have installed it and run the cells , i keep getting this error!any ideas,please help

Fadwa Fawzy • Posted on Latest Version • 2 months ago • Options 0

You didn't subtract the imagenet mean from your data! Why? From what I have read, vgg16 has pre-processing steps, and we have to apply the same pre-processing to the input images

```
image[:,:,0] -= 103.939

image[:,:,1] -= 116.779
image[:,:,2] -= 123.68
image /= 255.
image = image.transpose((2,0,1))
image = np.expand_dims(image, axis=0)
```

kechan • Posted on Latest Version • 2 months ago • Options 0

I got hit with: URL fetch failure on https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5: None -- [Errno -2] Name or service not known

Is the pre-trained weight still on GitHub?