VGG19 With First 5 Layers Frozen

Reference:https://www.kaggle.com/atrisaxena/keras-plant-seedlings-vgg19-augmentation

In [1]:
```
import warnings
warnings.filterwarnings('ignore')
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
print(os.listdir("../input"))
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"

import cv2
from keras.preprocessing.image import ImageDataGenerator
from PIL import Image
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import ImageGrid
import numpy as np
from keras.utils import np_utils
from keras import applications
from keras.preprocessing.image import ImageDataGenerator
from keras import optimizers
from keras.models import Sequential, Model
from keras.layers import Dropout, Flatten, Dense, GlobalAveragePooling2D
from keras.callbacks import ModelCheckpoint, LearningRateScheduler, TensorBoard, EarlyStopping
```

```
['plant-seedlings-classification', 'withvalid']


Using TensorFlow backend.
```

In [2]:
```
CATEGORIES = ['Black-grass', 'Charlock', 'Cleavers', 'Common Chickweed', 'Common wheat', 'Fat H
en', 'Loose Silky-bent',
              'Maize', 'Scentless Mayweed', 'Shepherds Purse', 'Small-flowered Cranesbill', 'Su
gar beet']
```

```
NUM_CATEGORIES = len(CATEGORIES)
```

In [3]:
```
SEED = 123
data_dir = '../input/withvalid'
train_dir = os.path.join(data_dir, 'train_valid/train_valid/train')
valid_dir = os.path.join(data_dir, 'train_valid/train_valid/valid/valid')
test_dir = os.path.join(data_dir, 'test/test')
```

Number of training images for each Category

In [4]:
```
for category in CATEGORIES:
    print('{} {} images'.format(category, len(os.listdir(os.path.join(train_dir, category)))))
```

```
Black-grass 237 images
Charlock 351 images
Cleavers 258 images
Common Chickweed 550 images
Common wheat 199 images
Fat Hen 428 images
Loose Silky-bent 589 images
Maize 199 images
Scentless Mayweed 465 images
Shepherds Purse 208 images
Small-flowered Cranesbill 447 images
Sugar beet 347 images
```

In [5]:
```
for category in CATEGORIES:
    print('{} {} images'.format(category, len(os.listdir(os.path.join(valid_dir, category)))))
```

```
Black-grass 26 images
Charlock 39 images
Cleavers 29 images
Common Chickweed 61 images
Common wheat 22 images
Fat Hen 47 images
Loose Silky-bent 65 images
Maize 22 images
Scentless Mayweed 51 images
Shepherds Purse 23 images
Small-flowered Cranesbill 49 images
Sugar beet 38 images
```

In [6]:
```
train = []
for category_id, category in enumerate(CATEGORIES):
    for file in os.listdir(os.path.join(train_dir, category)):
        train.append(['train/{}/{}'.format(category, file), category_id, category])
train = pd.DataFrame(train, columns=['file', 'category_id', 'category'])
train.head(2)
train.shape
```

Out[6]:

	file	category_id	category
0	train/Black-grass/d3c72d4c3.png	0	Black-grass
1	train/Black-grass/37d85d833.png	0	Black-grass

Out[6]:
```
(4278, 3)
```

In [7]:
```
valid = []
for cat_id, cat in enumerate(CATEGORIES):
    for f in os.listdir(os.path.join(valid_dir, cat)):
        valid.append(['valid/{}/{}'.format(cat, file), cat_id, cat])
valid = pd.DataFrame(valid, columns=['file', 'category_id', 'category'])
valid.head(5)
```

Out[7]:

	file	category_id	category
0	valid/Black-grass/dc39e01ec.png	0	Black-grass
1	valid/Black-grass/dc39e01ec.png	0	Black-grass
2	valid/Black-grass/dc39e01ec.png	0	Black-grass
3	valid/Black-grass/dc39e01ec.png	0	Black-grass
4	valid/Black-grass/dc39e01ec.png	0	Black-grass

In [8]:
```
test = []
for file in os.listdir(test_dir):
    test.append(['test/{}'.format(file), file])
test = pd.DataFrame(test, columns=['filepath', 'file'])
test.head(2)
test.shape
```

Out[8]:

	filepath	file
0	test/0e8492cb1.png	0e8492cb1.png
1	test/b687160f5.png	b687160f5.png

Out[8]:

```
(794, 2)
```

See some of the Images

In [9]:

```
fig = plt.figure(1, figsize=(NUM_CATEGORIES, NUM_CATEGORIES))
grid = ImageGrid(fig, 111, nrows_ncols=(NUM_CATEGORIES, NUM_CATEGORIES), axes_pad=0.05)
i = 0
for category_id, category in enumerate(CATEGORIES):
    for filepath in train[train['category'] == category]['file'].values[:NUM_CATEGORIES]:
        ax = grid[i]
        img = Image.open("../input/withvalid/train_valid/train_valid/"+filepath)
        img = img.resize((240,240))
        ax.imshow(img)
        ax.axis('off')
        if i % NUM_CATEGORIES == NUM_CATEGORIES - 1:
            ax.text(250, 112, filepath.split('/')[1], verticalalignment='center')
        i += 1
plt.show();
```

In [10]:

```
model = applications.VGG19(weights = "imagenet", include_top=False, input_shape = (240, 240, 3)
)
for layer in model.layers[:5]:
    layer.trainable = False

x = model.output
x = Flatten()(x)
x = Dense(1024, activation="relu")(x)
x = Dropout(0.5)(x)
x = Dense(1024, activation="relu")(x)
predictions = Dense(12, activation="softmax")(x)

model_final = Model(input = model.input, output = predictions)

model_final.compile(loss = "categorical_crossentropy", optimizer = optimizers.SGD(lr=0.0001, mo
mentum=0.9), metrics=["accuracy"])
model_final.summary()
```

```
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.1/
vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5
80142336/80134624 [==============================] - 1s 0us/step
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 240, 240, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 240, 240, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 240, 240, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 120, 120, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 120, 120, 128)     73856
```

```
block4_conv1 (Conv2D)        (None, 30, 30, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 30, 30, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 30, 30, 512)       2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, 30, 30, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 15, 15, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 15, 15, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 15, 15, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 15, 15, 512)       2359808
_________________________________________________________________
block5_conv4 (Conv2D)        (None, 15, 15, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 25088)             0
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              25691136
_________________________________________________________________
dropout_1 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 1024)              1049600
_________________________________________________________________
dense_3 (Dense)              (None, 12)                12300
=================================================================
Total params: 46,777,420
Trainable params: 46,664,844
Non-trainable params: 112,576
_________________________________________________________________
```

In [11]:
```
gen = ImageDataGenerator(
            rotation_range=360.,
            width_shift_range=0.3,
            height_shift_range=0.3,
            zoom_range=0.3,
            horizontal_flip=True,
            vertical_flip=True)
```

In [12]:
```
train_data_dir = "../input/withvalid/train_valid/train_valid/train"
train_generator = gen.flow_from_directory(
                        train_data_dir,
                        target_size = (240, 240),
                        batch_size = 16,
                        class_mode = "categorical")
```

```
Found 4278 images belonging to 12 classes.
```

In [13]:
```
valid_data_dir = "../input/withvalid/train_valid/train_valid/valid/valid"
valid_generator = gen.flow_from_directory(
                        valid_data_dir,
                        target_size = (240, 240),
                        batch_size = 16,
                        class_mode = "categorical")
```

```
Found 472 images belonging to 12 classes.
```

In [14]:
```
checkpoint = ModelCheckpoint("vgg16_withvalid.h5", monitor='loss', verbose=1, save_best_only=Tr
ue, save_weights_only=False, mode='auto', period=1)
early = EarlyStopping(monitor='loss', min_delta=0, patience=10, verbose=1, mode='auto')
```

Train our Model

In [15]:
```
batchsize = 16
STEP_SIZE_TRAIN=train_generator.n/batchsize
STEP_SIZE_VALID=valid_generator.n/batchsize

model_final.fit_generator(
                    generator=train_generator,
                    validation_data=valid_generator,
                    steps_per_epoch=STEP_SIZE_TRAIN,
                    validation_steps=STEP_SIZE_VALID,
                    epochs = 50,
                    shuffle= True,
                    callbacks = [checkpoint, early])
```

```
Epoch 1/50
268/267 [==============================] - 183s 682ms/step - loss: 2.5749 - acc: 0.1203 - val_
loss: 2.4212 - val_acc: 0.1547

Epoch 00001: loss improved from inf to 2.57495, saving model to vgg16_withvalid.h5
Epoch 2/50
268/267 [==============================] - 160s 597ms/step - loss: 2.4085 - acc: 0.1374 - val_
loss: 2.3803 - val_acc: 0.1398

Epoch 00002: loss improved from 2.57495 to 2.40887, saving model to vgg16_withvalid.h5
Epoch 3/50
```

```
Epoch 3/50
268/267 [==============================] - 160s 599ms/step - loss: 2.3778 - acc: 0.1574 - val_
loss: 2.3057 - val_acc: 0.1737

Epoch 00003: loss improved from 2.40887 to 2.37705, saving model to vgg16_withvalid.h5
Epoch 4/50
268/267 [==============================] - 160s 596ms/step - loss: 2.3057 - acc: 0.1796 - val_
loss: 2.3438 - val_acc: 0.1589

Epoch 00004: loss improved from 2.37705 to 2.30502, saving model to vgg16_withvalid.h5
Epoch 5/50
268/267 [==============================] - 159s 592ms/step - loss: 2.0990 - acc: 0.2636 - val_
loss: 1.7064 - val_acc: 0.3983

Epoch 00005: loss improved from 2.30502 to 2.09897, saving model to vgg16_withvalid.h5
Epoch 6/50
268/267 [==============================] - 160s 595ms/step - loss: 1.7016 - acc: 0.3932 - val_
loss: 1.3945 - val_acc: 0.4619

Epoch 00006: loss improved from 2.09897 to 1.70268, saving model to vgg16_withvalid.h5
Epoch 7/50
268/267 [==============================] - 158s 590ms/step - loss: 1.3843 - acc: 0.4969 - val_
loss: 1.0433 - val_acc: 0.6568

Epoch 00007: loss improved from 1.70268 to 1.38378, saving model to vgg16_withvalid.h5
Epoch 8/50
268/267 [==============================] - 158s 591ms/step - loss: 1.0669 - acc: 0.6164 - val_
loss: 0.7816 - val_acc: 0.7564

Epoch 00008: loss improved from 1.38378 to 1.06813, saving model to vgg16_withvalid.h5
Epoch 9/50
268/267 [==============================] - 159s 594ms/step - loss: 0.8954 - acc: 0.6847 - val_
loss: 0.8958 - val_acc: 0.6992
```

```
Epoch 00009: loss improved from 1.06813 to 0.89684, saving model to vgg16_withvalid.h5
Epoch 10/50
268/267 [==============================] - 160s 598ms/step - loss: 0.7021 - acc: 0.7487 - val_
loss: 0.5431 - val_acc: 0.8114

Epoch 00010: loss improved from 0.89684 to 0.70257, saving model to vgg16_withvalid.h5
Epoch 11/50
268/267 [==============================] - 161s 602ms/step - loss: 0.5842 - acc: 0.7937 - val_
loss: 0.4194 - val_acc: 0.8665

Epoch 00011: loss improved from 0.70257 to 0.58469, saving model to vgg16_withvalid.h5
Epoch 12/50
268/267 [==============================] - 160s 597ms/step - loss: 0.4928 - acc: 0.8235 - val_
loss: 0.4025 - val_acc: 0.8644

Epoch 00012: loss improved from 0.58469 to 0.49374, saving model to vgg16_withvalid.h5
Epoch 13/50
268/267 [==============================] - 160s 598ms/step - loss: 0.4506 - acc: 0.8389 - val_
loss: 0.3796 - val_acc: 0.8750

Epoch 00013: loss improved from 0.49374 to 0.45061, saving model to vgg16_withvalid.h5
Epoch 14/50
268/267 [==============================] - 160s 596ms/step - loss: 0.4100 - acc: 0.8553 - val_
loss: 0.4577 - val_acc: 0.8581

Epoch 00014: loss improved from 0.45061 to 0.41028, saving model to vgg16_withvalid.h5
Epoch 15/50
268/267 [==============================] - 160s 597ms/step - loss: 0.3682 - acc: 0.8671 - val_
loss: 0.2887 - val_acc: 0.8983

Epoch 00015: loss improved from 0.41028 to 0.36859, saving model to vgg16_withvalid.h5
Epoch 16/50
268/267 [==============================] - 160s 597ms/step - loss: 0.3434 - acc: 0.8832 - val_
loss: 0.3457 - val_acc: 0.8750
```

.....

```
Epoch 00038: loss did not improve from 0.14950
Epoch 39/50
268/267 [==============================] - 159s 594ms/step - loss: 0.1532 - acc: 0.9459 - val_
loss: 0.2726 - val_acc: 0.9089

Epoch 00039: loss did not improve from 0.14950
Epoch 40/50
268/267 [==============================] - 160s 597ms/step - loss: 0.1425 - acc: 0.9459 - val_
loss: 0.1947 - val_acc: 0.9322

Epoch 00040: loss improved from 0.14950 to 0.14275, saving model to vgg16_withvalid.h5
Epoch 41/50
268/267 [==============================] - 167s 622ms/step - loss: 0.1340 - acc: 0.9525 - val_
loss: 0.1875 - val_acc: 0.9470

Epoch 00041: loss improved from 0.14275 to 0.13313, saving model to vgg16_withvalid.h5
Epoch 42/50
268/267 [==============================] - 160s 596ms/step - loss: 0.1549 - acc: 0.9448 - val_
loss: 0.2178 - val_acc: 0.9386

Epoch 00042: loss did not improve from 0.13313
Epoch 43/50
268/267 [==============================] - 160s 597ms/step - loss: 0.1392 - acc: 0.9499 - val_
loss: 0.2089 - val_acc: 0.9301

Epoch 00043: loss did not improve from 0.13313
Epoch 44/50
268/267 [==============================] - 159s 594ms/step - loss: 0.1508 - acc: 0.9473 - val_
loss: 0.1819 - val_acc: 0.9470

Epoch 00044: loss did not improve from 0.13313
Epoch 45/50
 50/267 [====>.........................] - ETA: 1:53 - loss: 0.1530 - acc: 0.9450
```

In [18]:

```
prediction = []
for f in test['file']:
    img = cv2.imread(os.path.join(test_dir,f))
    img = cv2.resize(img,(240,240))
    img = np.asarray(img)
    img = img.reshape(1,240,240,3)
    pred = model_final.predict(img)
    prediction.append(classes.get(pred.argmax(axis=-1)[0])) #Invert Mapping helps to map Label
```

In [19]:

```
pred = pd.DataFrame({'species': prediction})
test =test.join(pred)
```

Final submission File

In [20]:

```
test.to_csv('submission.csv', index=False)
```

In [21]:

```
test.head()
```

Out[21]:

	filepath	file	species
0	test/0e8492cb1.png	0e8492cb1.png	Sugar beet
1	test/b687160f5.png	b687160f5.png	Small-flowered Cranesbill
2	test/fea3da57c.png	fea3da57c.png	Sugar beet
3	test/f6d250856.png	f6d250856.png	Shepherds Purse
4	test/e73e308be.png	e73e308be.png	Fat Hen

VGG19+Xception

The only difference between the combination of pre-trained models and VGG19 with first 5 layers frozen is the model structure as following:

In [8]:

```
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.layers import BatchNormalization
scale=299

model = applications.VGG19(weights = "imagenet", include_top=False, input_shape = (scale, scale
, 3))
add_model=applications.Xception(weights = "imagenet", include_top=False, input_shape = (scale,
scale, 3))

model = Sequential()
model.add(add_model)
model.add(Conv2D(20, kernel_size=(3, 3), activation='relu',input_shape=(scale, scale, 3)))
model.add(BatchNormalization(axis=3))
model.add(Conv2D(20, kernel_size=(3, 3), activation='relu'))
model.add(BatchNormalization(axis=3))
model.add(MaxPooling2D(pool_size=(3, 3), strides=(2, 2)))
model.add(Dropout(0.2))


model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation = 'relu'))
model.add(Dropout(0.5))
model.add(Dense(12, activation='softmax'))


#compling and show model
model.compile(loss = "categorical_crossentropy", optimizer = optimizers.SGD(lr=0.0001, momentum
=0.9), metrics=["accuracy"])
model.summary()
```

```
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.1/
vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5
80142336/80134624 [==============================] - 1s 0us/step
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.4/
xception_weights_tf_dim_ordering_tf_kernels_notop.h5
83689472/83683744 [==============================] - 5s 0us/step
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
xception (Model)             (None, 10, 10, 2048)      20861480
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 8, 8, 20)          368660
_________________________________________________________________
batch_normalization_5 (Batch (None, 8, 8, 20)          80
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 6, 6, 20)          3620
_________________________________________________________________
batch_normalization_6 (Batch (None, 6, 6, 20)          80
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 2, 2, 20)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 2, 2, 20)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 80)                0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               20736
_________________________________________________________________
dropout_2 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                16448
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 12)                780
_________________________________________________________________
dense_3 (Dense)              (None, 12)                780
=================================================================
Total params: 21,271,884
Trainable params: 21,217,276
Non-trainable params: 54,608
_________________________________________________________________
```