Below is a step by step tutorial on how I achieved image augmentation using keras in my spyder ide. I wanted to explore my options by also using Tensorflow but for object detection, and I encountered a problem with my pycocotools refusing to install, I didn’t ponder too much into it as I should but I’ll still get back to it someday. Now, back to the situationship we have.

What is data or image augmentation, it is useful if you would like to increase the amount of training of validation data or augment your data. If you have 2000 images and would like to augment it to get 5000 or 10,000 of those images, data augmentation can be useful but if you only have about 5 or 10 images, don’t expect to get 2000 or 20000 out of data augmentation and still be able to get decent results out of your deeplearning because the deeplearning is going to be bias towards these 5 or 10 images that you are augmenting. If you have less than 100 or 200 images, it is recommended that your use the traditional machine learning method. This is because you can extract features and use random forest or support vector machines.

I’ll break it into two three phases for you to easily understand and follow up. Each phase has its own steps, and I believe by the end of it all you’ll achieve image augmentation easily.

Phase one: Setup Python Virtual Environment in Spyder IDE

Step1: Install virtualenv package with pip. i.e pip install virtualenv, this can be easily done in the python command prompt or anaconda command prompt.

Step2: Create a virtual env for your project. i.e virtualenv projectname project name is whatever name you give your virtual environment, in my case I named mine object, i.e virtualenv object

Step3: Activate the env. i.e activate object

Step4: Configure Spyder IDE for your environment. You must have successfully installed Spyder from anaconda distribution, go to (<https://www.spyder-ide.org/>) to easily download it. On the Spyder IDE, go to main menu, click Tools, go to Preferences, select Python Interpreter. Click on the radio button beside Use the following Python Interpreter. In the open space there, browse to the python.exe file in the Scripts folder of your virtual environment folder, select the path C:\Users\esther\object\Scripts\python.exe for the python interpreter.

And we have successfully setup our Spyder IDE for image augmentation. Now we move to the next Phase

Phase two: This phase is about installing the necessary libraries involved.

Now back to our command prompt which we setup up from our virtual environment folder, we go to the path in the virtualenv folder and type ‘cmd’ to go to our command prompt. While here we can run the following commands:

pip install spyder-kernels

pip install tensorflow #keras requires tensorflow version 2.2 and above

pip install keras

pip install scikit-image #this is also known as skimage

pip install lxml

pip install pillow

pip install matplotlib

pip install contextlib2

pip install cython

pip install tf\_slim

From here, we are almost ready for image augmentation. Before we close the command prompt, go to the Scripts folder, i.e cd Scripts, when you enter the scripts folder directory in the command prompt, type activate, press enter. This will activate the batch file named activate in the Scripts folder. After that, type activate object which is the name of the virtual env created.

Then go to your Spyder IDE, open a new file which we will use for our data augmentation, ensure that the file is saved in the virtual environment folder, this will enable it run without any hiccup.

Phase three: Here we write our code for image augmentation to be carried out successfully.

In your script, type the following

import tensorflow

from keras.preprocessing.image import ImageDataGenerator

from skimage import io

#these are the most important modules to be used for image augmentation

NB: Ensure the folder containing the image(s) are also in the virtual environment folder.

datagen = ImageDataGenerator(

rotation\_range = 45, #i.e random rotation between 0 and 45 degrees.

width\_shift\_range = 0.2, #i.e shifting in x-axis of whatever the image size is by 20 percent

height\_shift\_range = 0.2, #doing the same thing in the y-axis

shear\_range = 0.2,

zoom\_range = 0.2, #zoom\_in, zoom\_out by 20 percent

horizontal\_flip = True,

fill\_mode = ‘constant’, cval = 125) #for fill\_mode also try nearest, constant and reflect to select your preference. For fill\_mode, when you move the images by 20 percent, there are some dark spaces that might be left over in your image, what will that be filled with. If the fill mode is “constant”, you’ll have to give a value, if you don’t give a value, it will be black pixels. If you put a value of cval, 0=black, 125 is grey, while 255 is white.

Now for us to read a single image, we can type

x = io.imread(‘C:\Users\esther\object\xyxyx.jpeg’) #you can use opencv to read the image but don’t forget to import it.

x=x.reshape((1, ) + x.shape) # array with shape (1, 256, 256, 3) 1= number of images, with 256 by 256 size and it’s a color imagethats why its 3. If it’s a grey image we’ll call it 1.