*Public LB 0.938 & Local LB 0.887* **Model summary:**

**Basic info.**

Network: se-resnext50

Batch-size: 320

Training set: 25k/class

Valid set: 80/class

Image size: 96\*96

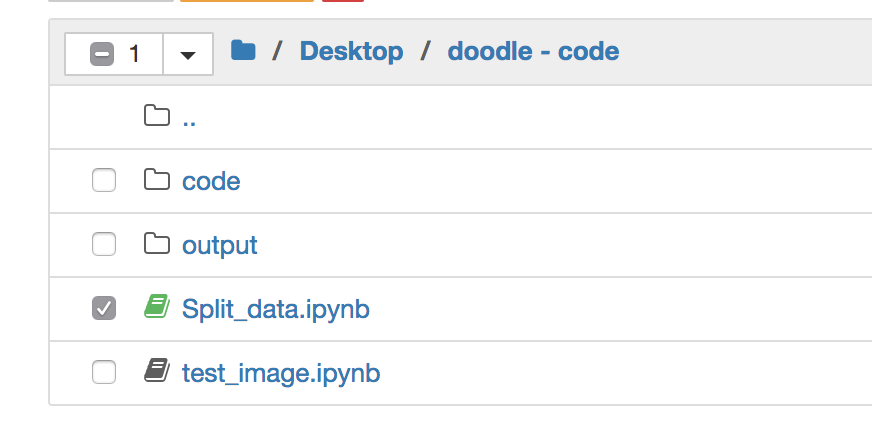
LR schedule: SGD (momentum=0.9, weight\_decay=0.0001), start from learning rate = 0.01, train until convergence, then change into 0.001, train again until convergence

**Tricks**

1. Encode image by different colors (in the sequence of stroke), it increases my result from *local 0.877 & public lb 0.926* to *local 0.885 & public lb 0.93*
2. Add data augmentation. Specifically, hflip under prob=0.5, and randomly crop 80% of image then resize for training set; center crop 80% of image then resize for valid (and test) set, then the result increases to *local 0.887 & public lb 0.935*
3. TTA. For each test image, do hflip × five different crops (center, up & left, bottom & right and so on) to get 10 images and predict respectively. It achieves *public lb 0.938*

***To use the code, please follow the following step:***

1. **download** training set (simplified version)
2. **Preparing dataset**. Split dataset, keep one valid set (80 images/class) and one holdout set (80 images/class too)

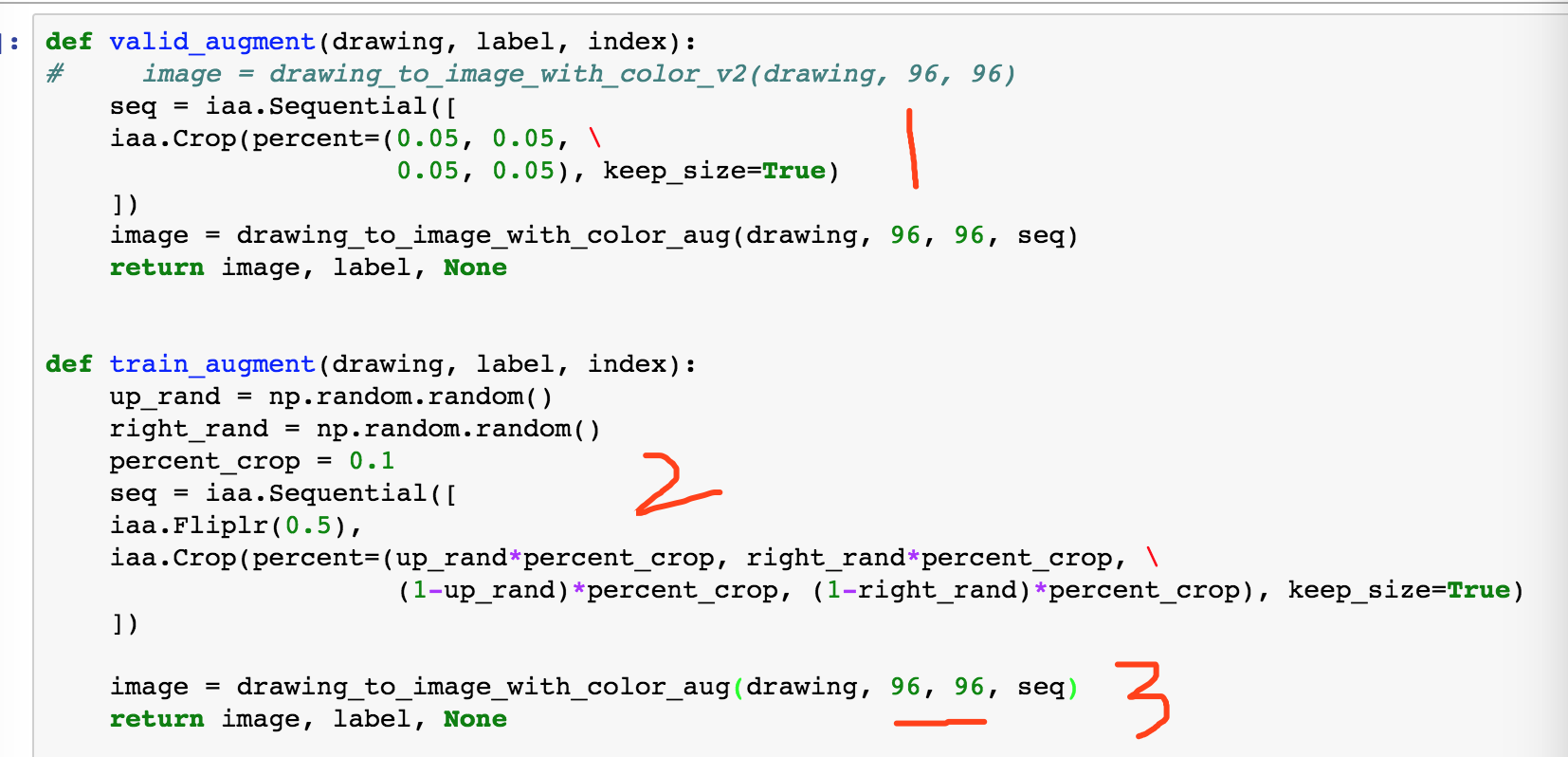


* Open *Split\_data.ipynb*
* Please change the path into the place you put your training set

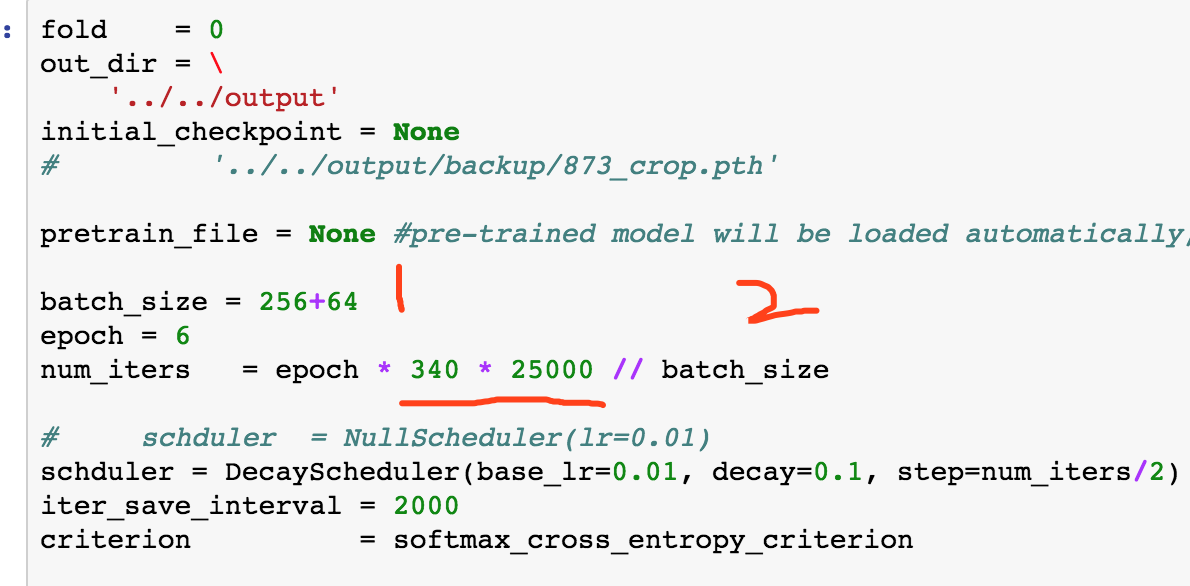


* Output all data (*X\_keep* in code) or sample data (*X\_sample*) for training (all data will definitely increase our results)
* Finally, run code/process/ data\_process.ipynb, nothing need to change.

1. **Training**. In code/train/train.ipynb, and set your configurations:



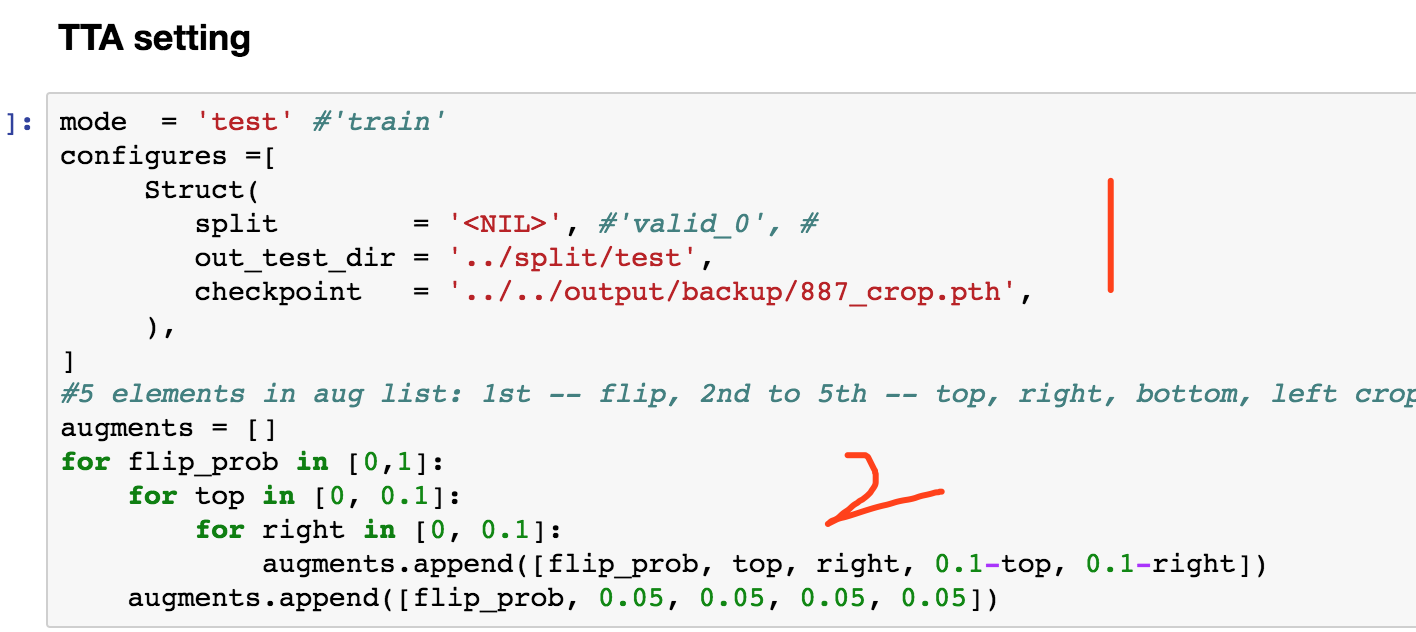
* You may change the augmentation strategies, here “1” means do center crop for valid set; “2” means do hflip (prob=0.5) and random crop (totally 10% for left&right, and 10% for up and bottom); “3” means input image size is 96\*96. Maybe 128 or 256 is better?



* Change your batch\_size and epoch in “1”, larger batch\_size seems better. As for the epoch, here I use 6 and under my decay scheduler, it will train 3 epoch with lr = 0.01, and 3 epoch with lr = 0.001. **Whereas**, as I said, you should keep training under lr = 0.01 until your model is convergent. Here, 6 is just my rough estimation.

As shown in “2”, “340 \* 25000” should be replaced by the training set size you use.

1. **Predicting**. Move to code/ local\_submit-withTTA.ipynb, in step 3, the training log and checkpoints are stored in output/



* Choose the weights you need for prediction in “1”
* Change the TTA strategies in “2” if you wish. My TTA function is ugly - -, I just input an array to define the corresponding augmentation.