[PersonAttrubutes121.ipynb](https://github.com/Shivdutta/EIP4/blob/master/Assignment5/PersonAttrubutes121.ipynb" \o "PersonAttrubutes121.ipynb)

1. The image data is loaded in google drive, unzipped and loaded in dataframe.
2. The annotation data is loaded in dataframe.
3. The Cut-out function 'get\_random\_eraser' copied from github. Cut-out is a data augmentation technique which we will use in data generator.
4. Multilabel classification is done using One Hot Encoding.
5. Persondatagenerator is a data generator will feed the images to model based on batch size.
6. Following are the target/output variable which are created inside Persondatagenerator:
   1. Gender
   2. Image Quality
   3. Age
   4. Weight
   5. Bag
   6. Pose
   7. Footwear
   8. Emotion
7. Train test split is done with ratio of 80:20.
8. Random State =42 is set to prevent data leak
9. Image from resized folder was loaded to test the image.
10. The data frame namely train\_gen and valid\_gen were created for datagenerators to train and validation data respectively with required image data augmentation, image rescaling and image normalization

**Approach1:**

1. **Architecture**: The configurable RESNET layer mentioned in Keras site was used.
2. The Stages and Residual blocks are configurable in the architecture. The receptive field was calculated using

[https://fomoro.com/research/article/receptive-field-calculator#](https://fomoro.com/research/article/receptive-field-calculator)

1. The calculated receptive field is uploaded in the git: Receptive Field Calculator.xls. This is done to check whether 200\*200 images converge fully with the architecture.

All the calculation were done on Conv2D

1. There are 6 stages and 3 residual block in the architecture
2. The identity layer is added initial(res=0) layer and for stages to prevent the loss and pass early detected features.
3. Inside resent block, instead of 2D Convolution, the depth wise convolution is used. This is done to prevent the reduce the number of parameters and increase the receptive field.
4. The model is printed for layout and summary.
5. The number of parameter compared to Conv2D is reduced.
6. Thereafter the backbone is created which is basically the resent layer with 18/20/32 vertebrates.
7. This is fed to neck which is performs flattening(1D) and mixing using dense layer to create a fully connected layer.
8. From each of neck 8 head comprising of 8 features are created. Before creating head, tower which acts as connector to head is created. Each parameterized tower performs activations and mixing using dense layer.
9. The model is compiled. The model requires Stochastic Gradient Descent as optimizer, with other hyper parameters like momentum, learning rate, nesterov and decay.
10. Different loss weights are adjusted on multiple iterations and penalizing wherever required to crease the accuracy.
11. Start and Stop approach is used to train model with 50 epochs. This approach was taken because to avoid losing the state of model because of connectivity to google cloud and internet.
12. The training and validation accuracy are calculated at the end of each 50 epochs.
13. Next training for 50 epochs and similarly training and validation accuracy are calculated at the end of each 50 epochs.
14. Thereafter 4 iterations of 25 epochs are run. The training and validation accuracy are calculated at the end of each 25 epochs.
15. The logs are embedded along with python notebook and the below is consolidated accuracy:

{'age\_output\_acc': 0.800000011920929,

'age\_output\_loss': 0.45471820838394617,

'bag\_output\_acc': 0.7216022007522129,

'bag\_output\_loss': 0.5328253230878285,

'emotion\_output\_acc': 0.8537946428571429,

'emotion\_output\_loss': 0.38752382301858496,

'footwear\_output\_acc': 0.7275545824141729,

'footwear\_output\_loss': 0.5337334853552637,

'gender\_output\_acc': 0.6990327380952381,

'gender\_output\_loss': 0.9930417569620269,

'image\_quality\_output\_acc': 0.7082093443189349,

'image\_quality\_output\_loss': 0.5692407322071847,

'loss': 116.55328905014764,

'pose\_output\_acc': 0.7433035927159446,

'pose\_output\_loss': 0.5466791980323338,

'weight\_output\_acc': 0.8231026785714286,

'weight\_output\_loss': 0.41636188328266144}

[PersonAttrubutes125.ipynb](https://github.com/Shivdutta/EIP4/blob/master/Assignment5/PersonAttrubutes121.ipynb" \o "PersonAttrubutes121.ipynb)

**Approach2:**

1. Resnet 50 block is used .
2. One cycle test is used to train with triangular,triangular2 and exp\_range mode.
3. The learning rate is determined using the range test with different set of values. Enclosed the LRFinder.docx. LRFinder for Keras implementation is used.
4. The learning rate with maximum accuracy and near to linear slope is used for training.
5. Model is trained with Triangular, Triangular2 and Exponential Range.
6. It is observed that validation loss is reducing and accuracy is improving. While approaching the end in epoch 25, validation loss is almost 0 which tells network has learned.
7. The logs are embedded along with python notebook and the below is consolidated accuracy:

180/180 [==============================] - 24s 132ms/step

{'age\_output\_acc': 0.9067708333333333,

'age\_output\_loss': 0.49414551109075544,

'bag\_output\_acc': 0.9324652777777778,

'bag\_output\_loss': 0.2633275344967842,

'emotion\_output\_acc': 0.9052083333333333,

'emotion\_output\_loss': 0.305016254964802,

'footwear\_output\_acc': 0.9671875,

'footwear\_output\_loss': 0.20339321601721977,

'gender\_output\_acc': 0.9855034722222222,

'gender\_output\_loss': 0.11283879623644882,

'image\_quality\_output\_acc': 0.9050347222222223,

'image\_quality\_output\_loss': 0.39112594591246713,

'loss': 2.3462664763132732,

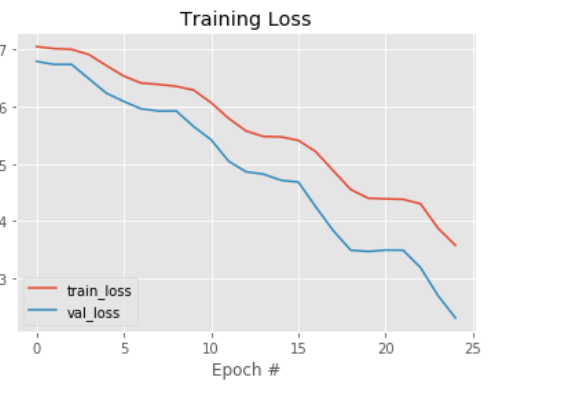
'pose\_output\_acc': 0.9782118055555555,

'pose\_output\_loss': 0.19471510607335302,

'weight\_output\_acc': 0.8861979166666667,

'weight\_output\_loss': 0.3817040988140636}

1. Below graph depicts the training accuracy is continuously reducing:



1. Below graph depicts the cyclical learning rate during model training:

