Name(s) : Ayaan Pupala, Maciej Kryziak
Kaggle Team Name: Maciej-Ayaan2

Part-1A: Pre-designed network for multi-label classification

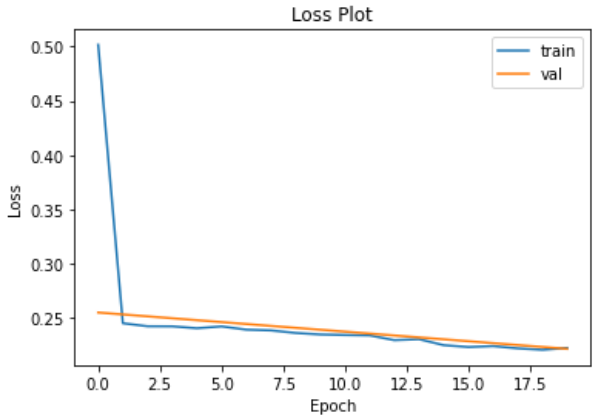
In this part, you will practice to train a neural network both by training from scratch or fine-tuning.

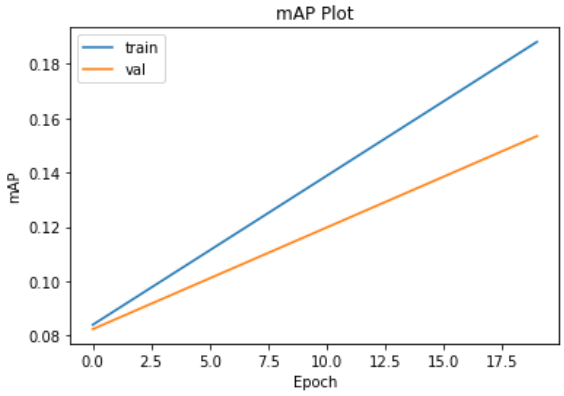
MP3_P1_Introduction.ipynb in your assignment3_p1_starterkit should provide you with enough instruction to start with.

We are asking you to provide the following results.

Due to the notebook of part A running extremely slowly these models were run in the notebook for part B. I have included the outputs in the submissions in separate pdfs of the partB notebook.

1. Simple Classifier
    a.Report test mAP for simple classifier: 0.1548
    b.Visualize loss and mAP plots:

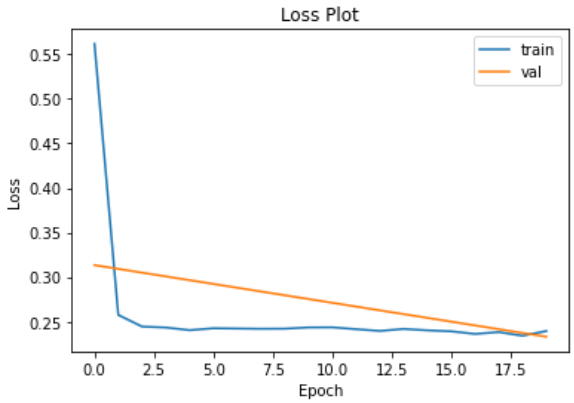


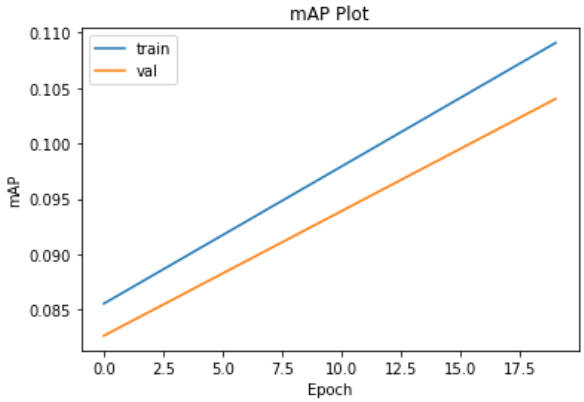


    c.Provide analysis (at least 3 sentences): We can see that the training loss decreases fairly quickly until 2 epochs and then reduces quite slowly along with the validation loss.The mAp on the training set increases much faster than the

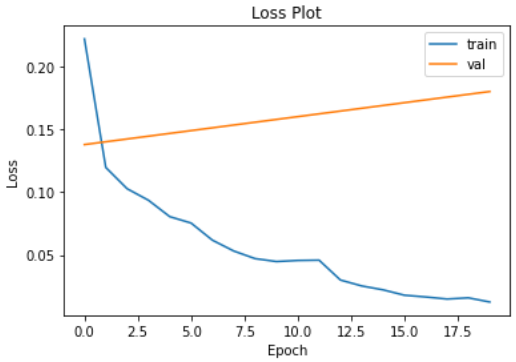
validation set. SInce we used this as a baseline i think that this model was overfitting the training data

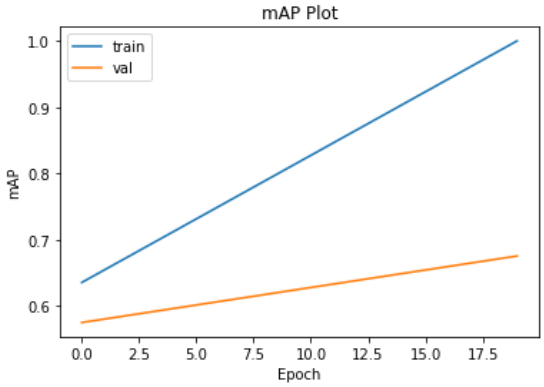
2. AlexNet from Scratch
    a.Report test mAP for alexnet: probably around 0.0988
    b.Visualize loss and mAP plots:





3. Pretrained AlexNet
    Report test mAP for pretrained alexnet: 0.6858
    Visualize loss and mAP plots





Provide analysis on differences to training from scratch (at least 3 sentences):
The pretrained Alexnet has a much higher accuracy from the start. The validation loss seems to increase for the pretrained alexnet. The pretrained Alexnets training and validation mAp are quite far apart but the from scratch one is much closer.

Part-1B: Self designed network for multi-label classification

MP3_P1_Develop_Classifier in your assignment3_p1_starterkit should provide you with enough instruction to start with. You upload your output of your self-designed network to kaggle.

Did you upload final CSV file on Kaggle: **Yes**

1. My best mAP on Kaggle: 0.36577 (1- mAP = 0.63423)
2. Factors which helped improve my model
    a. Data augmentation by transform RandomResizedCrop
    b. Adding Batch Normalization after each convolutional layer
    c. Running for 300 epochs
3. Table for final architecture (replace below with your best architecture design):

Layer No.	Layer Type	Kernel size (for conv layers)	Input | Output dimension	Input | Output Channels (for conv layers)
1	conv2d	5	64	3 | 5
2	BatchNorm		64	
3	relu	-	64|64	-
4	maxpool2d		64	-
5	conv2d	3	64|32	64 | 32
6	BatchNorm		32	
7	relu	-	32|32	-
8	maxpool2d		32	-
9	conv2d	3	32|16	32|16
10	BatchNorm		16	
11	relu		16|16	
12	maxpool2d		16	
13	linear	-	16*26*26 | 120	-
14	relu		120|120	
15	linear		120|84	
16	relu	-	84|84	-
17	linear	-	84| 20	-

The initial network provided to you can be considered as the BaseNet. A very important part of deep learning is understanding the ablation studies of various networks. So we would like you to do a few experiments. Note, this **doesn't need to be very exhaustive** and can be in a cumulative manner in an order you might prefer. Fill in the following table :

BaseNet used : SimpleClassifier

Serial #	Model architecture	Best mAP on test set
1	BaseNet	0.17
2	BaseNet + Batch Norm	0.274
3	BaseNet + Batch Norm + CosineAnnealingLR	0.2510
4	BaseNet + Batch Norm + step_lr	0.14
5	BaseNet + BatchNorm+ Dropout(0.2)	0.2869
6	BaseNet + BatchNorm + RandomHorizflip	0.2803
7	BaseNet + BatchNorm + RandomResizedCrop	0.2826
8	BaseNet + BatchNorm + Dropout(0.2) RandomResizedCrop+300 epochs	0.365

Make some analysis on why and how you think certain changes helped or didn't help:

I think the main issue we were having was that the model was overfitting the training data, we can see how this was the case as the 2 methods that reduce and combat overfitting helped us the most ie Batch normalization and dropout. We were not sure why learning rate scheduling did not improve our results at all and initially we weren't getting a difference when we ran for longer epochs. After implementing the random resized crop we started to see improvements as we ran longer and therefore we tried running for 300 epochs to see if we could hit the bench mark and we did.