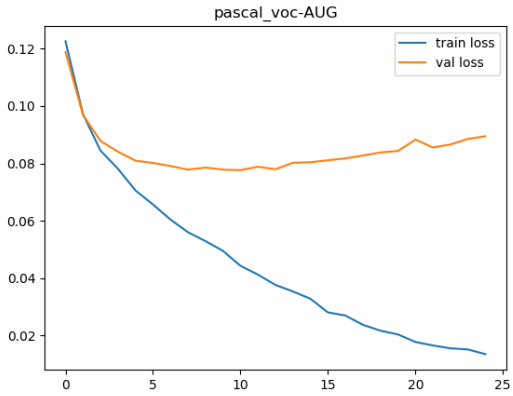
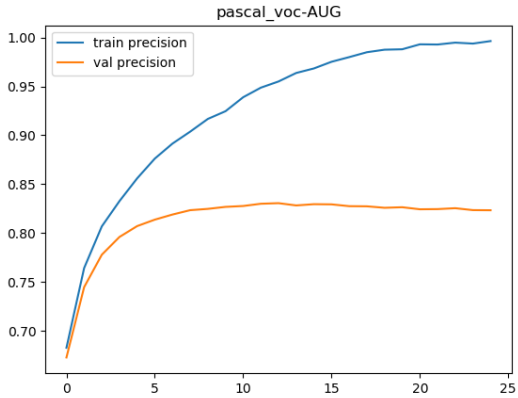
Deep Learning project Pascal VOC
Gary Ong 1002758

Model: Pretrained resnet18
Last layer FC (512,20)
Loss: BCE with logits loss
Optimizer: SGD Lr = 0.005
Scheduler: None
Image size 224x224 Plain resize no crop
Epoch: 25
Batch size: 16
No augmentation, ImageNet normalization numbers
MAP Score 0.8242

+ Random Horizontal Flip
+ Random Erase
MAP Score 0.8306

Loss /precision against epoch for model



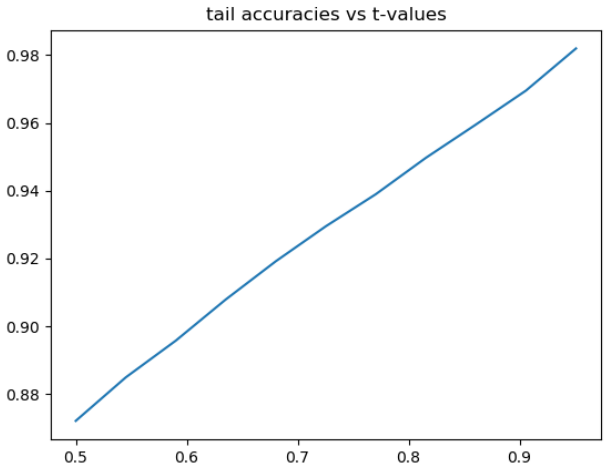


Model overfits quickly. Best score from validation set used for evaluation.

To see best and worst images go to the folder BestAndWorst5.
Here are the scores for MAP for each class and final macro average.

class	MAP
aeroplane	0.967
bicycle	0.862
bird	0.938
boat	0.872
bottle	0.597
bus	0.929
car	0.823
cat	0.954
chair	0.74
cow	0.759
diningtable	0.667
dog	0.904
horse	0.87
motorbike	0.892
person	0.955
pottedplant	0.602
sheep	0.856
sofa	0.624
train	0.943
tvmonitor	0.861
macro avg all	0.831

Macro averaged graph



Tail accuracies of each class at each t values.
Maximum t is taken from min of max (fx) of each class.

t-values	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	diningtable	dog	horse	motorbike	person	pottedplant	sheep	sofa	train	tvmonitor	average
0.5	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87
0.545091396	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
0.590182792	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
0.635274188	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91
0.680365584	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92
0.72545698	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93	0.93
0.770548376	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94
0.815639772	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
0.860731168	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96
0.905822564	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97
0.95091396	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98

How to run code:
To train the model run **train_model.py**
The first few lines contain some stuff in CAPS that you might want to change.
Produces saved model and loss precision graphs in directory.

To evaluate the model run **eval_model.py**
The first few lines contain some stuff in CAPS that you might want to change.
Produces csv file for MAP for each class and macro averaged and csv file for tail accuracies.
Also produces 5 classes with 5 best and 5 worst images.

All functions required written in **pascal_functions.py.** Designed to be run independently.
Dataset **PacalVocDataset.py** modified from starter code.

Training code takes about 3 min per epoch on laptop GTX 1050ti
Testing code takes about 1 min on laptop GTX 1050ti.