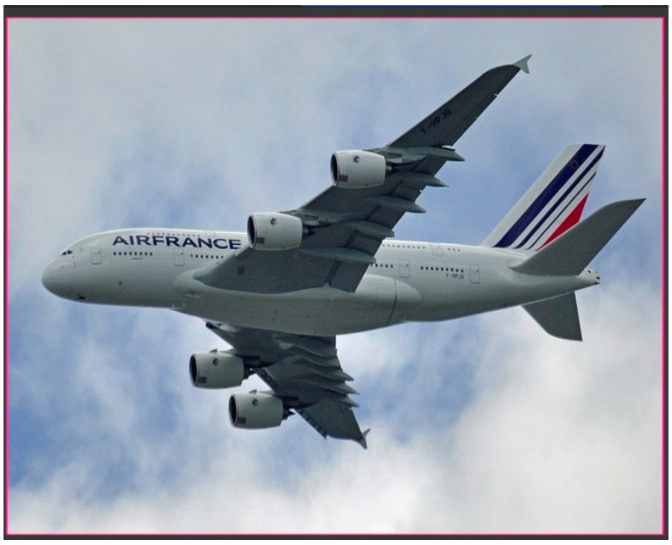
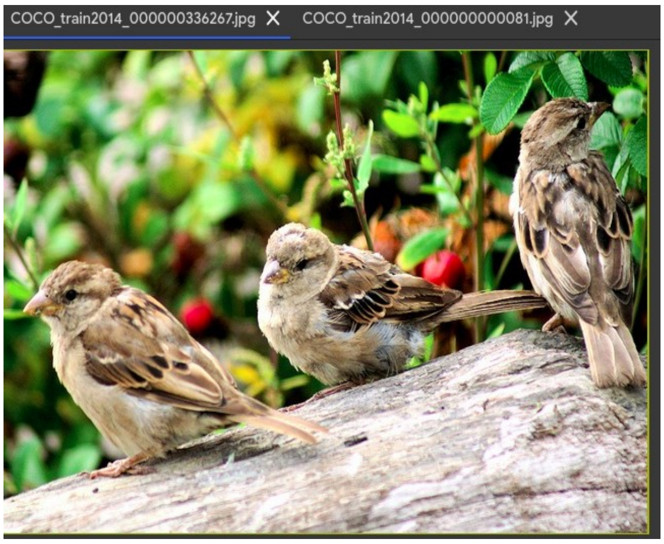
Name : Panneerselvam N
Role : Research Intern – Inspect
Date : 02/07/2021

Evaluation of Custom model:

In Today morning , i had tried to evaluate yesterday's trained checkpoints . But the object is not detected . Because while training the loss remains same , and returns Nan value also.




COCO_train2014_000000336267.jpg
COCO_train2014_000000000081.jpg

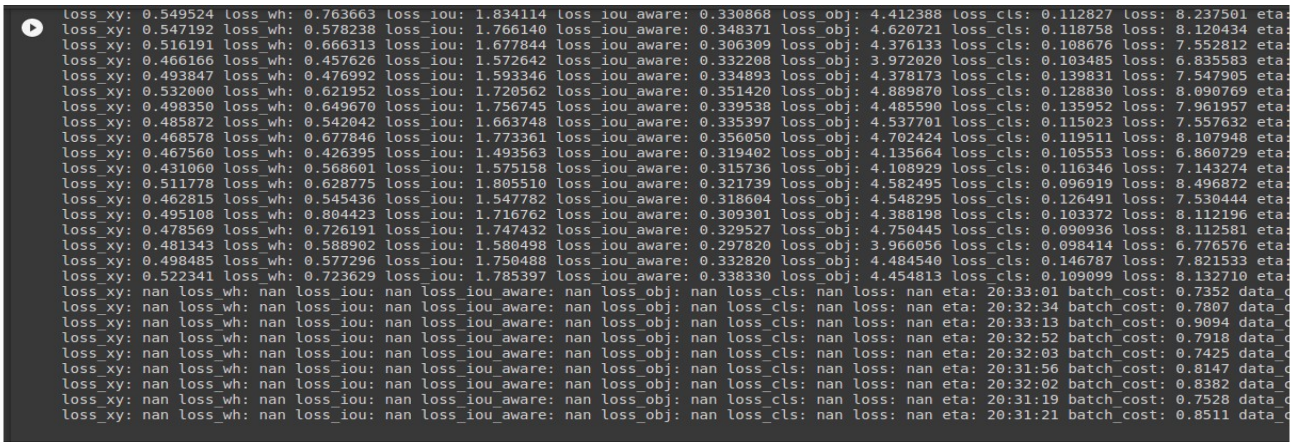
Above two images is output of custom model , object is not detected.

So, i had tried to fix Nan loss.

Tried Methods :
method 1:
I had converted json file again and verified information is correct or not.

Method 2:
Yesterday i had used opencv's imread and imwrite to copy image file from drive to my work location . While doing that , in default it stores in BGR format. So , i had changed BGR to RGB and copied files again .

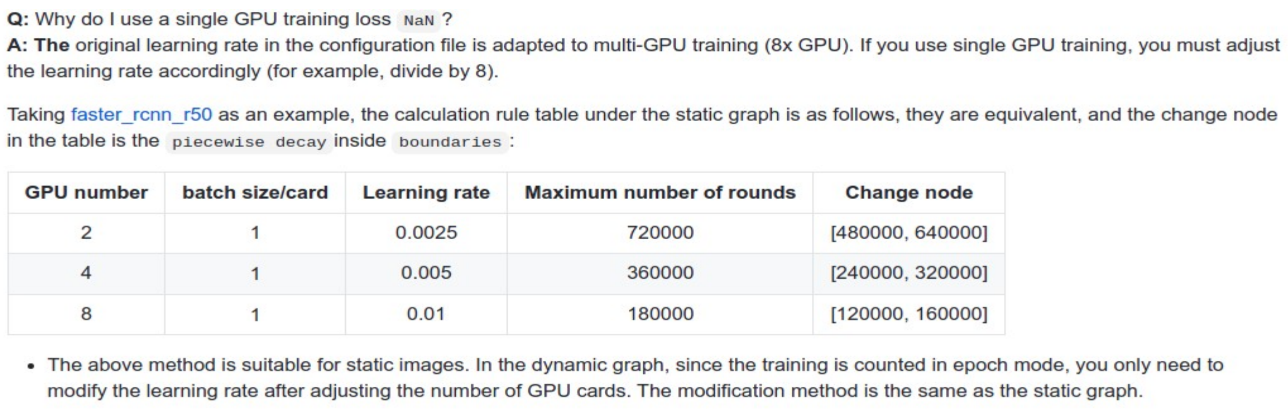
After modifying above two things , again tried to train my model. But error is not fixed .



```
loss_xy: 0.549524 loss_wh: 0.763663 loss_iou: 1.834114 loss_iou_aware: 0.330868 loss_obj: 4.412388 loss_cls: 0.112827 loss: 8.237501 eta:
loss_xy: 0.547192 loss_wh: 0.578238 loss_iou: 1.766140 loss_iou_aware: 0.348371 loss_obj: 4.620721 loss_cls: 0.118758 loss: 8.120434 eta:
loss_xy: 0.516191 loss_wh: 0.666313 loss_iou: 1.677844 loss_iou_aware: 0.306309 loss_obj: 4.376133 loss_cls: 0.108676 loss: 7.552812 eta:
loss_xy: 0.466166 loss_wh: 0.457626 loss_iou: 1.572642 loss_iou_aware: 0.332208 loss_obj: 3.972020 loss_cls: 0.103485 loss: 6.835583 eta:
loss_xy: 0.493847 loss_wh: 0.476992 loss_iou: 1.593346 loss_iou_aware: 0.334893 loss_obj: 4.378173 loss_cls: 0.139831 loss: 7.547905 eta:
loss_xy: 0.532000 loss_wh: 0.621952 loss_iou: 1.720562 loss_iou_aware: 0.351420 loss_obj: 4.889870 loss_cls: 0.128830 loss: 8.090769 eta:
loss_xy: 0.498350 loss_wh: 0.649670 loss_iou: 1.756745 loss_iou_aware: 0.339538 loss_obj: 4.485590 loss_cls: 0.135952 loss: 7.961957 eta:
loss_xy: 0.485872 loss_wh: 0.542042 loss_iou: 1.663748 loss_iou_aware: 0.335397 loss_obj: 4.537701 loss_cls: 0.115023 loss: 7.557632 eta:
loss_xy: 0.468578 loss_wh: 0.677846 loss_iou: 1.773361 loss_iou_aware: 0.356050 loss_obj: 4.702424 loss_cls: 0.119511 loss: 8.107948 eta:
loss_xy: 0.467560 loss_wh: 0.426395 loss_iou: 1.493563 loss_iou_aware: 0.319402 loss_obj: 4.135664 loss_cls: 0.105553 loss: 6.860729 eta:
loss_xy: 0.431060 loss_wh: 0.568601 loss_iou: 1.575158 loss_iou_aware: 0.315736 loss_obj: 4.108929 loss_cls: 0.116346 loss: 7.143274 eta:
loss_xy: 0.511778 loss_wh: 0.628775 loss_iou: 1.805510 loss_iou_aware: 0.321739 loss_obj: 4.582495 loss_cls: 0.096919 loss: 8.496872 eta:
loss_xy: 0.462815 loss_wh: 0.545436 loss_iou: 1.547782 loss_iou_aware: 0.318604 loss_obj: 4.548295 loss_cls: 0.126491 loss: 7.530444 eta:
loss_xy: 0.495108 loss_wh: 0.804423 loss_iou: 1.716762 loss_iou_aware: 0.309301 loss_obj: 4.388198 loss_cls: 0.103372 loss: 8.112196 eta:
loss_xy: 0.478569 loss_wh: 0.726191 loss_iou: 1.747432 loss_iou_aware: 0.329527 loss_obj: 4.750445 loss_cls: 0.090936 loss: 8.112581 eta:
loss_xy: 0.481343 loss_wh: 0.588902 loss_iou: 1.580498 loss_iou_aware: 0.297820 loss_obj: 3.966056 loss_cls: 0.098414 loss: 6.776576 eta:
loss_xy: 0.498485 loss_wh: 0.577296 loss_iou: 1.750488 loss_iou_aware: 0.332820 loss_obj: 4.484540 loss_cls: 0.146787 loss: 7.821533 eta:
loss_xy: 0.522341 loss_wh: 0.723629 loss_iou: 1.785397 loss_iou_aware: 0.338330 loss_obj: 4.454813 loss_cls: 0.109099 loss: 8.132710 eta:
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:33:01 batch_cost: 0.7352 data_c
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:32:34 batch_cost: 0.7807 data_c
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:33:13 batch_cost: 0.9094 data_c
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:32:52 batch_cost: 0.7918 data_c
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:32:03 batch_cost: 0.7425 data_c
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:31:56 batch_cost: 0.8147 data_c
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:32:02 batch_cost: 0.8382 data_c
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:31:19 batch_cost: 0.7528 data_c
loss_xy: nan loss_wh: nan loss_iou: nan loss_iou_aware: nan loss_obj: nan loss_cls: nan loss: nan eta: 20:31:21 batch_cost: 0.8511 data_c
```

Solution :

Finally I had found the solution from official documentation .



Q: Why do I use a single GPU training loss `NaN` ?
A: The original learning rate in the configuration file is adapted to multi-GPU training (8x GPU). If you use single GPU training, you must adjust the learning rate accordingly (for example, divide by 8).

Taking faster_rcnn_r50 as an example, the calculation rule table under the static graph is as follows, they are equivalent, and the change node in the table is the `piecewise decay` inside `boundaries` :

GPU number	batch size/card	Learning rate	Maximum number of rounds	Change node
2	1	0.0025	720000	[480000, 640000]
4	1	0.005	360000	[240000, 320000]
8	1	0.01	180000	[120000, 160000]

- The above method is suitable for static images. In the dynamic graph, since the training is counted in epoch mode, you only need to modify the learning rate after adjusting the number of GPU cards. The modification method is the same as the static graph.

Problem is due to learning rate configurations , we need to divide the default learning rate by 8 .

After above modification ,tried to train my model ,but GPU limits are over. So , I will check it tomorrow and update it.