BLG435E - Assignment #2

Ramazan Yetişmiş-150190708

January 2, 2021

State Representation

In this part I get help from the sources [1] [3].

No Tail

For this type I have 12 states.And this states firs divided to groups.Firs one is **Danger**,second one is **Move Direction** and the last one is the **Apple**. All these sets have contains 4 direction values **('U','D','L','R')**
1.Danger Check
For this states I checked if there is any danger around.For this operation I considered the one point distance check.The checking operation is simple. For instance if the snake's position is 'R' and if and the next right position is **wall** then **danger straight=1**.For the **NO TAIL mode danger is only the walls, because it has no body also for this reason we can check is there is a danger down because wall can be 1 point below.**
Danger states=('straight','right','left','down')⟶(0,0,1,0) means right side is **WALL**
2.Direction Check
Direction check is simple we just convert the letters into 4 binary numbers like **Direction('left','right','up,'down')=(0,1,0,0) ⟶ right**.
3.Apple Check
For this state we are simply looking for the apple position and compare it with current position.And again we are sending 4 four directions as binary format.
Apple('left','right','up,'down')=(0,1,0,0) ⟶ right.

TRON & CLASSIC

For this modes the state numbers are same 11 in my state representation.And also there are some minor changes from **NO TAIL** mode.
Difference One: Because this time snake has body so we have to check the body collision as well.

Difference Two:
The snake has body so have always danger in the back so we remove 'danger down' state. As a result we reduced 1 state and added extra body control to the Danger state.
Example State Representation For No Tail=[1,0,0,0,1,0,0,0,1,0,0,0] means Danger is ahead,direction is left, and food is on the left side.

Reward Function

For the reward function I do not have lots of criteria. For all the states I look at 4 conditions.

Eat Apple	Closer to Apple	away from Apple	Dies
1	0.1	-0.1	-1

Coding

No Tail

Training 1:

For the first training hyper parameters I used the initial given values.

Learning Rate	Batch Size	Max Capacity	Steps
1e-2	256	10000	35000

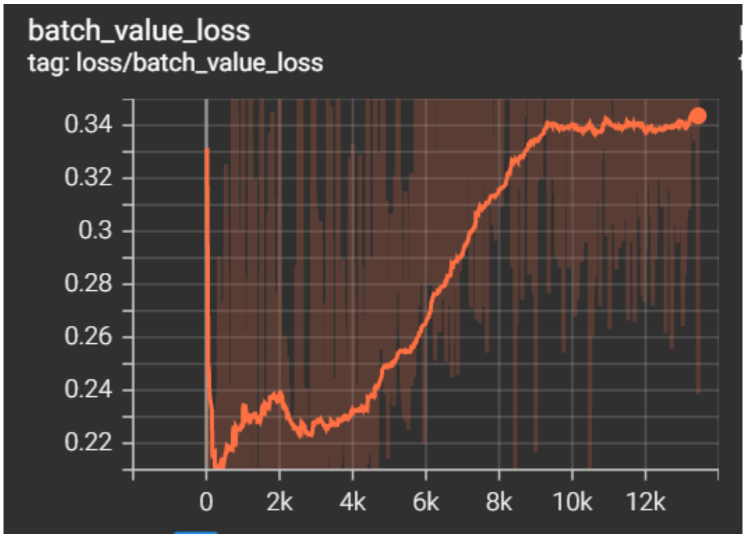

Figure 1: Batch loss function

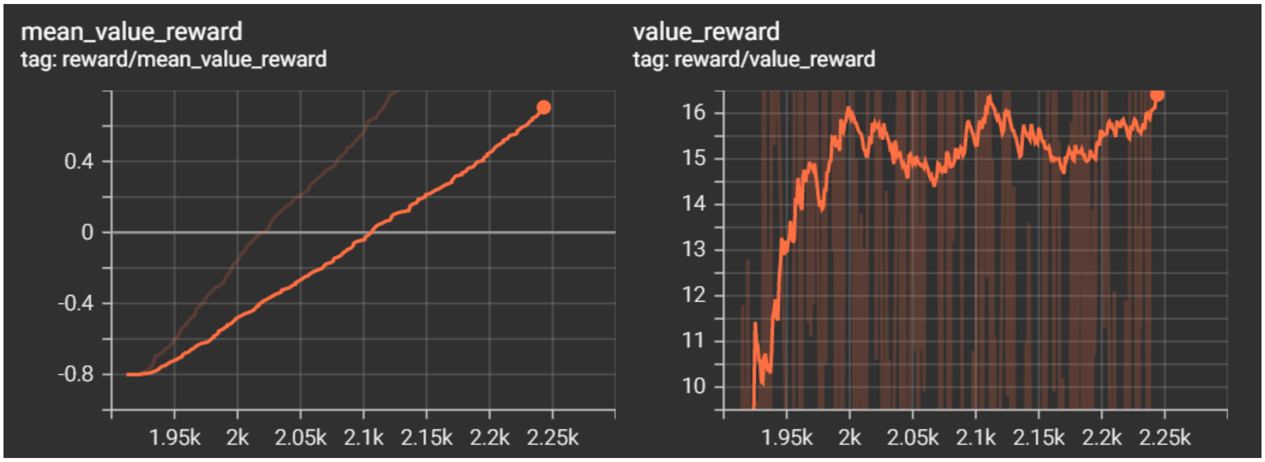

Figure 2: Reward Function

According to these parameters my loss function was not sufficient.Thus, Fırst thing I do is to make learning smaller.For the reward values it is increasing this means our agent is learning but it is not consistent at least not now.
Learning Rate:A too-high learning rate can lead the model to converge too rapidly to a poor solution, while a too-low learning rate can cause the process to stall.So for **No Tail** mode this learning rate was big so it converged to the sub local minimal point not global minimal.Now we have tor try smaller learning rate

Training 2:

Second training hyper parameters **Learning rate changed**.

Learning Rate	Batch Size	Max Capacity	steps
1e-3	256	10000	35000

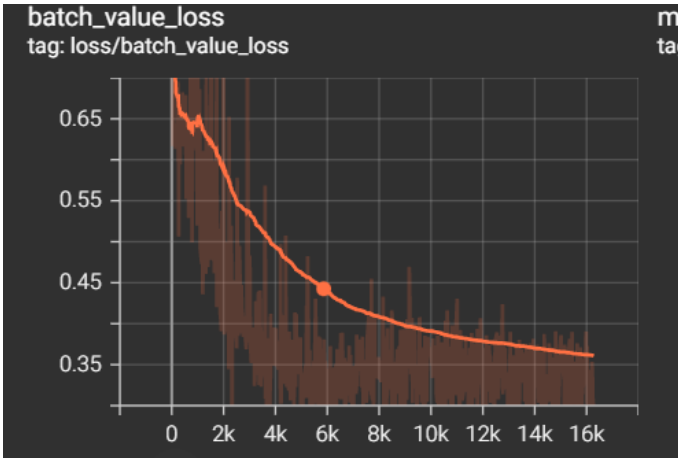

Figure 3: Loss Function lr= 1e-3 bs=256

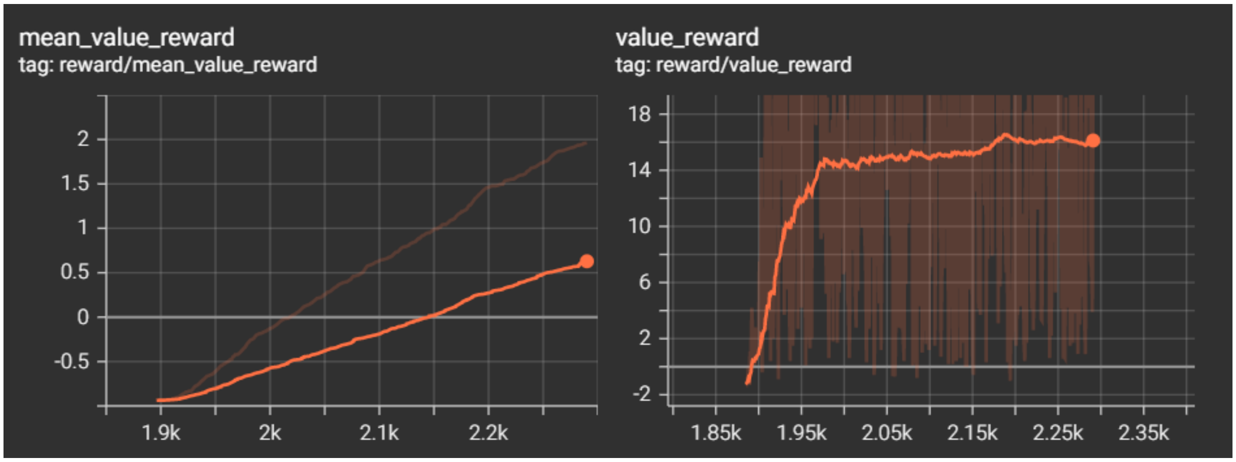

Figure 4: Reward Function lr= 1e-3 bs=256

As we can see when make learning rate smaller our loss function becomes smaller at every time step. That means our function will not find local minimum points.For the reward function we can say that there less fluctuations now it is more stable.Now we can change batch size.

Batch size: Smaller batch size leads to less accurate estimate of the gradient.Smaller batch takes up less memory. The total training technique takes less memory since the network is trained with fewer samples.So we have to find the best value for batch size.Because if it to small then there can be zigzag effect in the gradient descent.

Training 3:

In this step I changed the batch size to 1000.

Learning Rate	Batch Size	Max Capacity	steps
1e-3	1000	10000	35000

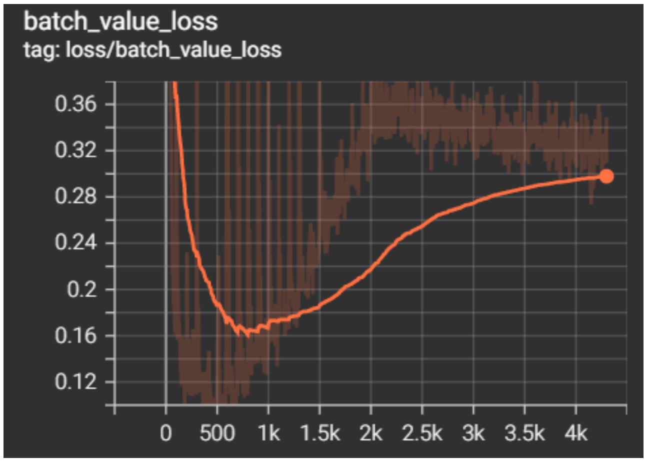

Figure 5: Loss Function lr= 1e-3 bs=1000

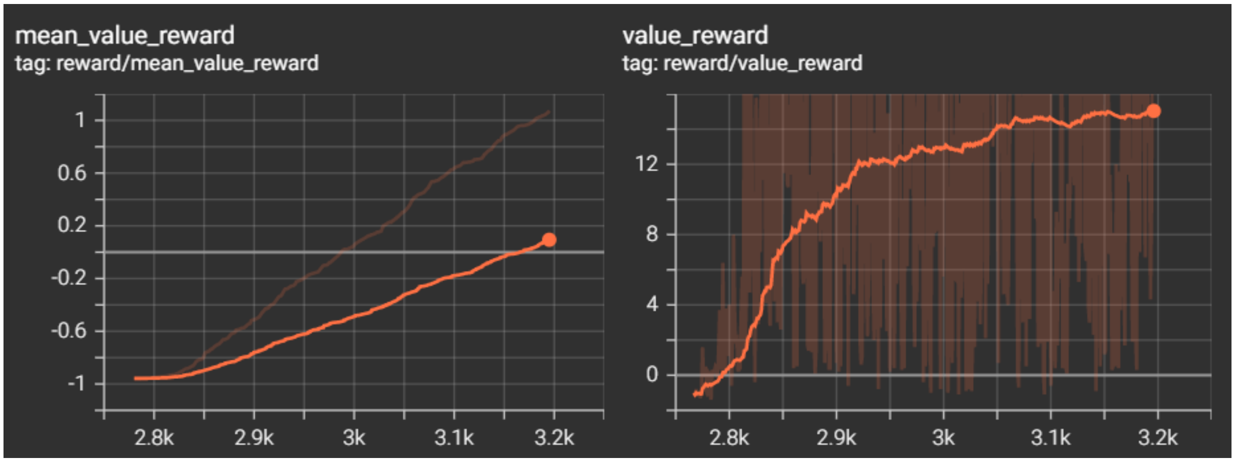

Figure 6: Reward Function lr= 1e-3 bs=1000

We can see that the loss function increase again after some point this is not the ideal case so increasing the batch value is not a proper move.For the reward function case we can easily say that it is again increasing but it is less than the BS=250 values.

Training 4:

In this training lr= 1e-4 and bs=500. I decreased the lr and bs.

Learning Rate	Batch Size	Max Capacity	steps
1e-4	500	10000	35000

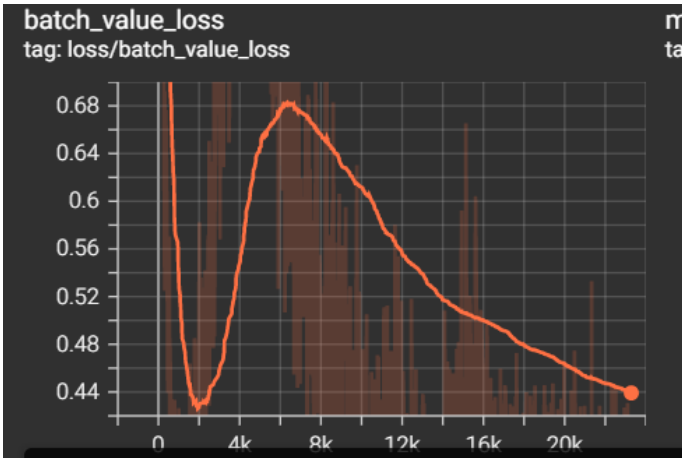

Figure 7: Loss Function lr= 1e-3 bs=1000

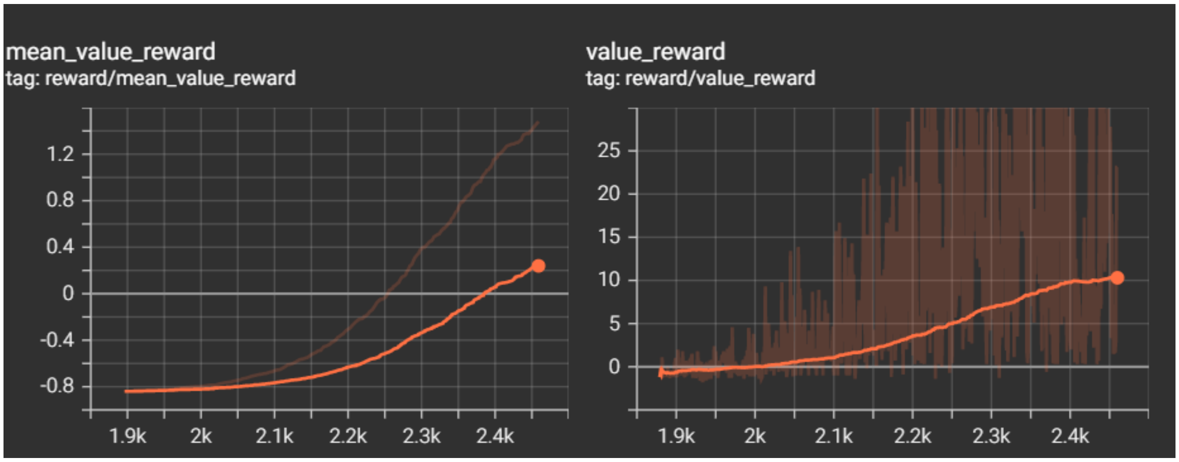

Figure 8: Reward Function lr= 1e-3 bs=1000

From these graphs that can be understand that if we increase the step number and batch size we can end up even better loss function. Because we can see from figure 7 loss continues to decrease.

Training 5:

In this training lr= 0.5*1e-3 and bs=256 and steps=50k.

Learning Rate	Batch Size	Max Capacity	steps
0.5*1e-3	256	10000	50000

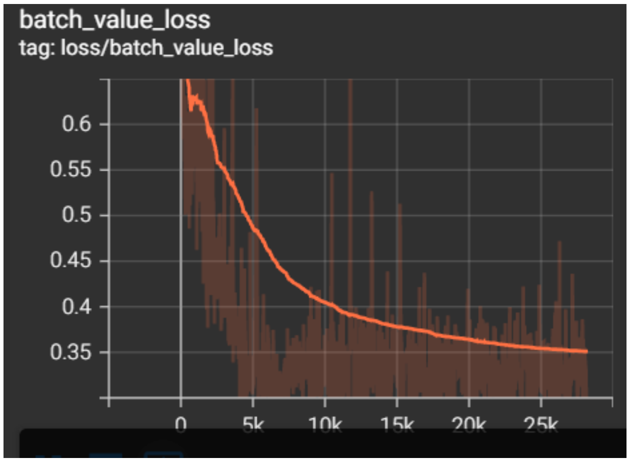

Figure 9: Loss Function lr= 0.5*1e-3 bs=256 steps=50k

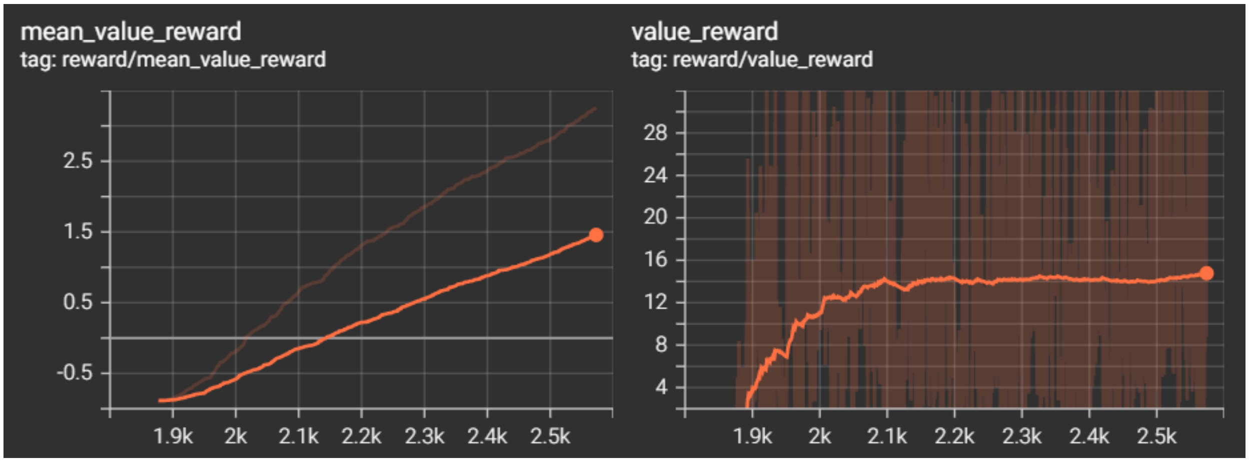

Figure 10: Reward Function lr= 0.5*1e-3 bs=256 steps=50k

Now our loss function seems good and reward is stably increasing.

Classic Mode

Now I trained my model according to the parameters with NOTAIL mode.Then lets analyse the results.

Training 6:

Learning Rate	Batch Size	Max Capacity	steps
0.5*1e-3	256	10000	50000

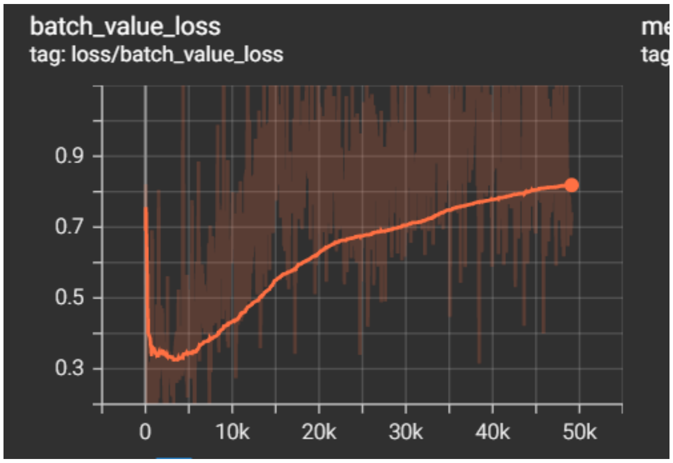

Figure 11: Loss Function lr= 0.5*1e-3 bs=256 steps=50k

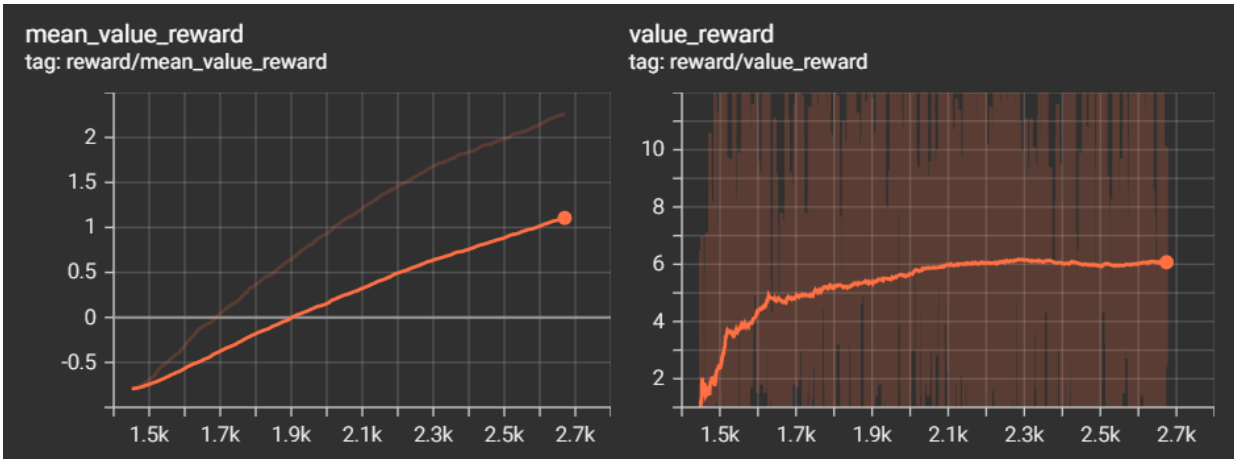

Figure 12: Reward Function lr= 0.5*1e-3 bs=256 steps=50k

For this mode our NO TAIL parameters worked fine but not very good because as we can see from the figure 11 loss stared to increase after some point (10k)we can apply early stopping in order to get rid of this problem. When we look at the reward function the reward values are stable and positive but not as much as the NO TAIL mode because the space is limited and the body of the snake increases whenever snake eats apple. Maybe when the snake is trained in a bigger play-field than the reward values may increase.So lets check if we can find better loss function.

Training 7:

Learning Rate	Batch Size	Max Capacity	steps
0.5*1e-4	128	10000	50000

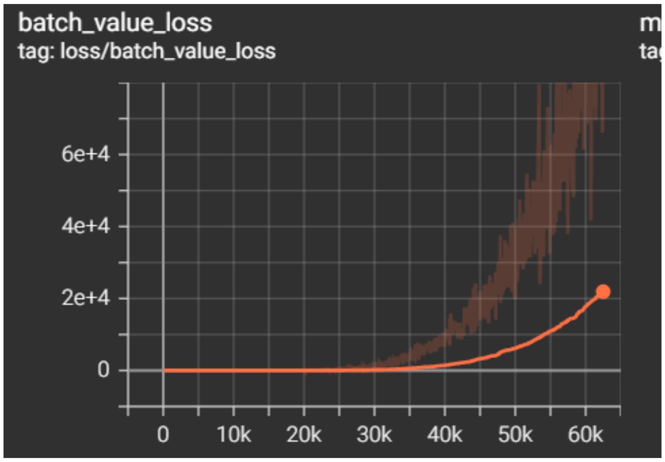

Figure 13: Loss Function lr= 0.5*1e-4 bs=128 steps=50k

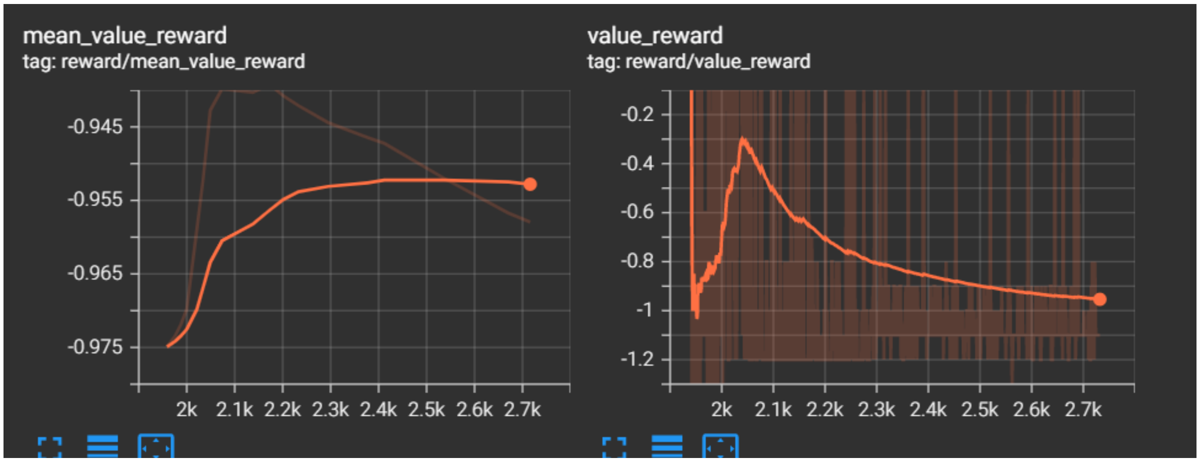

Figure 14: Reward Function lr= 0.5*1e-4 bs=128 steps=50k

With these hyper parameters the training process took longer than 3 hour so, I terminated the process and the reward and loss graph values are not suitable.

Training 8:

I just changed the capacity from 10000 to 15000.

Learning Rate	Batch Size	Max Capacity	steps
0.5*1e-3	256	15000	50000

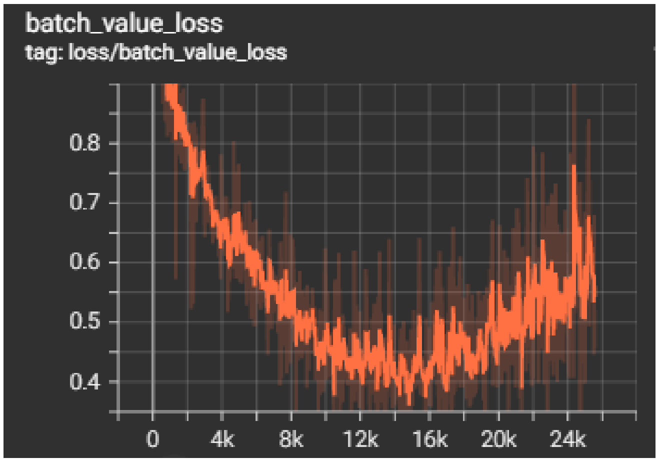

Figure 15: Loss Function lr= 0.5*1e-3 bs=256 steps=50k

Now we changed MAX_CAPACITY hyper parameter and it helped us to make effective

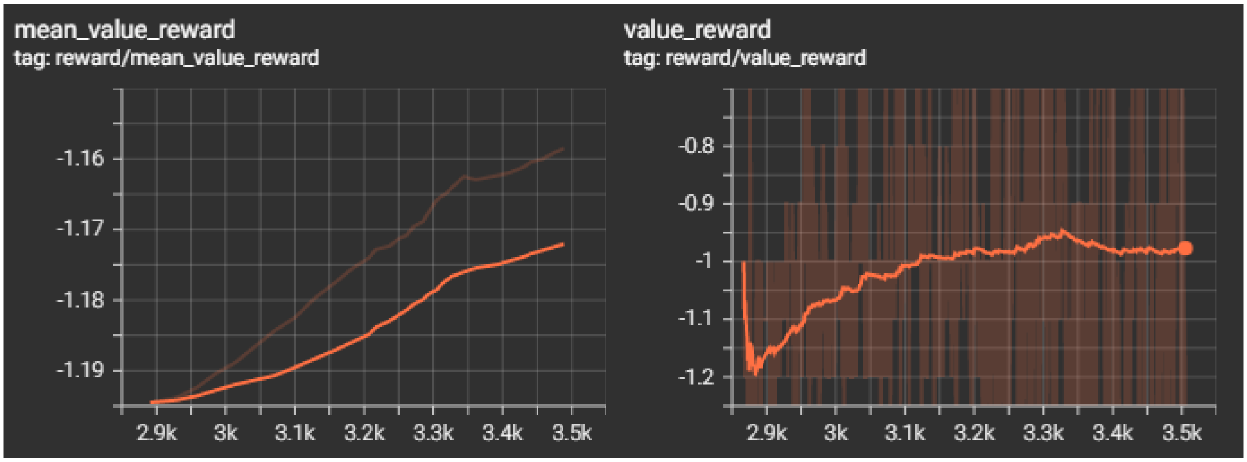

Figure 16: Reward Function lr= 0.5*1e-3 bs=256 steps=50k

loss function.

TRON MODE:

For this mode I used the same values with the classical mode

Learning Rate	Batch Size	Max Capacity	steps
0.5*1e-3	256	15000	50000

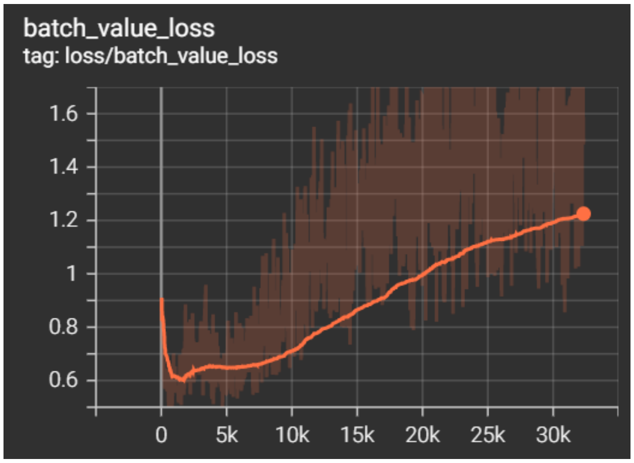

Figure 17: Loss Function lr= 0.5*1e-3 bs=256 steps=50k

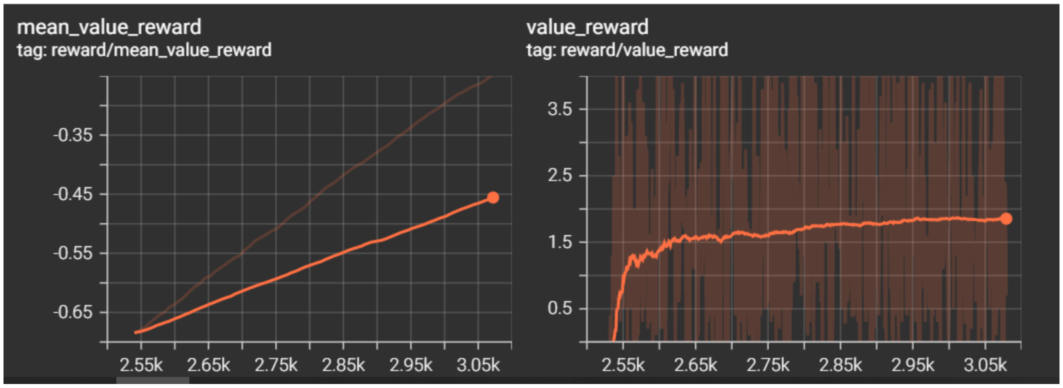

Figure 18: Reward Function lr= 0.5*1e-3 bs=256 steps=50k

For this type it took a lot longer to train the model with respect to the classical mode, and we can stop training the model earlier maybe around time step equals to the 10k because after that the loss began to increase immensely. The reason is the body of the snake increases faster than classical mode and the snake can not eat lots of apple and eventually after eating 2 or 3 apple it hits its own body. Reward function is consistent. The solution can be early stopping.

Instructions to compile/run your code

For plotting operations uncomment the written part in main.py and DQN.py. Also I used from torch.utils.tensorboard import SummaryWriter, so you also need to install that library.For this you can see the document [4].

References

1-https://towardsdatascience.com
2-1-https://github.com/sweetice
3-https://www.youtube.com/watch?v=PJl4iabBEz0&ab_channel=PythonEngineer
4-https://pytorch.org/tutorials/recipes/recipes/tensorboard_with_pytorch.html