3.1

The actor is the policy network, and the critic is the value network. Having an actor lets you optimize the policy of the agent. When combined with a critic, both networks can be optimized.

3.2

Line where action probabilities are calculated: 34 (logits = policy_network.policy(observation)).
Note that in the line above, the probabilites are still unnormalized; softmax is applied in line 37.
Line where an action is sampled based on the probabilities above: 36 (action = tf.random.categorical(logits, 1)[0][0]).
Line where the new state is generated based on the action: 47 (observation, r, done, info = env.step(action)).

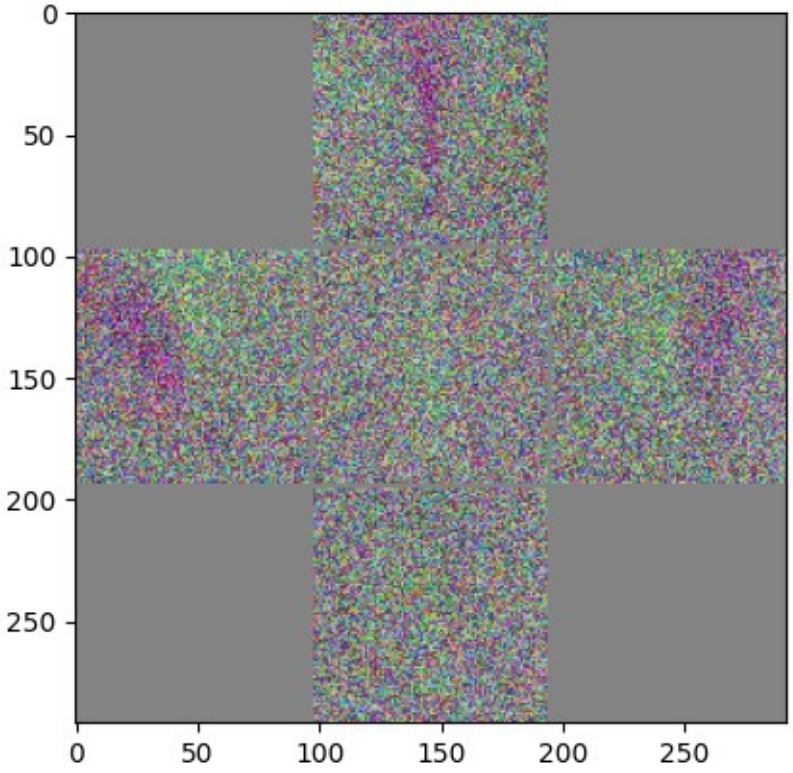
3.3

Since the observation tensor consists of a 96x96x3 image from which an optimal action has to be deduced, the optimal policy is very unlikely to be linear (the observation is too complex). For a linear policy to work as well as possible, I would imagine that the observation should be as non-complex as possible, for instance an image of a straight road.

3.4

A state with a high average return would likely be one where the car is on the road and still facing the road (i.e., it is not heading off the road). The agent would also have a high speed as this increases the likelihood that it completes the episodes in fewer frames, giving a higher reward (this can be observed in the observation through the control information bar). A state with a low value function could be any state in which the car is close to the edge of the playfield, in which it is likely that the agent will soon go out of it.

3.5



We can think the figure as a collection of heat maps, showing which weights are most prominent in deciding the probabilities of sampling each action. As we can see, the “left”, “right” and “gas” actions have many dark pixels forming lines in them, representing areas with influential weights. These darker areas are also placed respectively in the forward-left, forward-right, and forward-center parts of of each heat map.

An interpretation of this is analog to how a human driver would think instinctively when driving a car; when the driver observes the road going left further ahead, it is natural to want to turn left, resulting in the forward-left pixels being important when considering whether to turn left. The same logic applies to the “gas” and “right” action heat maps.

As for “straight” and “break”, they both look more like random noise that the other three, as if they hadn’t been trained. For “straight”, this might make sense, as going straight is the passive action of the five, and it can make most sense to choose if the observation doesn’t correspond particularly much to either of the patterns that the other heat maps respond to.

More interestingly, it seems the model has struggled to learn how to break. A human driver would obviously observe if the road ends or takes a sharp turn further on, but judging from the apparent randomness in what pixels the “break” weights deem most important, it seems such a pattern hasn’t been learned by the model.

3.6

N = 1:
min, max : (80.3135, 268.256)
median, mean : (147.486, 157.391)

Judging qualitatively, and having my own performance in mind when trying to compare to the agent’s, the agent looks well-trained. It drives at a high pace in straight lines with little turning, but is still able to clear the first sharp corner well, which requires very precise inputs and quick reactions from a human.

N = 2:
min, max : (111.673, 292.741)
median, mean : (182.743, 180.999)

The agent goes slightly quicker this time, meaning that acceleration is chosen somewhat more often. It seems a bit less stable, taking more turns even when the road Is in a straight line, and clearing corners with a bit more speed (the latter is actually a good thing, so long as the car doesn’t lose grip).

N = 4:
min, max : (74.0439, 270.837)
median, mean : (173.226, 178.054)

The agent goes even faster that for N = 2. It seems even less stable than before, taking even more turns when driving in straight lines, and doing so at even higher speeds. Still, it is able to maintain grip on the road, and seems to react faster and with more precise input than the vast majority of humans would be able to.

N = 8:
min, max : (51.1744, 258.884)
median, mean : (126.913, 133.239)

The agent goes even faster that for N = 4. At this point, the speed of the car is so high that the reaction speed of the agent looks vastly superior to what any human could achieve; it cleared the first corner before I managed to even observe the corner coming with my own eyes. I also noticed that when clearing that corner, it went off-track and through the grass a little bit, which didn’t happen for the other N sizes. It is unclear whether this just happened by randomness, or if the agent has actually learned to manipulate the physics of the environment to clear the corner more efficiently at higher speeds (which would be very impressive). However, going off-track may also be part of the explanation why N = 8 did worse score-wise than the other N sizes.